Visual Cross-view Geolocalization

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Florian Fervers

geb. in Mönchengladbach

Tag der mündlichen Prüfung	13. Dezember 2024
Hauptreferent	Prof. Dr.-Ing. Rainer Stiefelhagen Fakultät für Informatik Karlsruher Institut für Technologie
Korreferent	Prof. Dr. Bastian Leibe Visual Computing Institute Rheinisch-Westfälische Technische Hochschule Aachen

Florian Fervers
Visual Cross-view Geolocalization
Im Dezember 2024
Gutachter: Prof. Dr.-Ing. Rainer Stiefelhagen und Prof. Dr. Bastian Leibe

Karlsruher Institut für Technologie
Fakultät für Informatik
Institut für Anthropomatik und Robotik
Computer Vision for Human-Computer Interaction Lab

und

Fraunhofer Institut für Optronik, Systemtechnik und Bildauswertung
Abteilung für Objekterkennung

Abstract

Visual Geolocalization (VGL) aims to estimate the geolocation from which a photo is taken by matching it against a geo-registered model of the world, *e.g.* a database of reference images. VGL presents a potential alternative to Global Navigation Satellite Systems (GNSS) which require the availability of an external signal and offer only a limited accuracy of a few meters.

Research on VGL has long focused on using geo-registered street-view images from platforms such as Google Street-view as reference data. However, their sparse availability over different regions across the globe represents a significant limitation on both the scalability and cost-effectiveness of this approach.

Aerial imagery represents a possible alternative for the reference data against which street-view photos are localized. Their dense coverage of outdoor regions across the globe offers the potential to scale VGL to much larger regions and rival the sparse nature of street-view reference data.

However, this task represents a significant challenge due to the drastic change in viewpoint and scale between street-view and aerial perspective that methods have to overcome. It has consequently been labeled Cross-view Geolocalization (CVGL). Existing works rely on assumptions about the input data and controlled benchmark environments that reduce the complexity of the problem, but also severely limit their real-world applicability. They further focus on small search regions for which Street-view to Street-view Geolocalization (SVGL) is already widely adopted.

In this thesis, we revisit CVGL and provide novel methods, datasets and insight that push the boundaries of VGL in terms of scalability, accuracy and real-world applicability. We address the task via a decomposition into two independent sub-problems, *i.e.* retrieval and pose estimation, as follows.

To localize photos in large Regions of Interest (RoI), we propose a novel problem formulation for cross-view retrieval that partitions the RoI into *consistently sized* and *non-overlapping* geographical cells. Each cell represents a hypothesis for the camera location, and is assigned an embedding representation using aerial images at multiple Levels of Detail (LoD). A street-view photo is localized by retrieving the most similar cell in the embedding space via an efficient nearest neighbor approach.

Our work for the first time allows localizing street-view photos (1) in state-sized search regions such as Massachusetts with 23000km^2, (2) under real-world conditions with consumer-grade devices, (3) in the wild, *i.e.* without requiring information about the camera's intrinsics, lens distortion or orientation, and (4) without access to street-view images from the search region. For instance, our method localizes 60.6% of all non-panoramic photos uploaded to the crowd-sourcing platform Mapillary in the state of Massachusetts to within 50m of their ground-truth location.

To find the exact metric position and orientation of a camera around a prior pose estimate, we present a novel end-to-end trainable model that matches photos against a single aerial image to predict a probability distribution over possible poses on the image. We introduce a filtering framework that integrates the model's multi-modal predictions over time to estimate the long-term trajectory of a platform.

Our work for the first time allows determining the geo-registered ego-pose and long-term ego-trajectory of a platform (1) using only aerial imagery as reference database without access to street-view data from the test region, (2) in a purely vision-based manner without requiring range scanners such as lidar or radar, or external signals such as GNSS, and (3) with sub-meter accuracy. The method for instance achieves a median pose error of 0.87m on the Ford AV dataset, and a mean trajectory error of 0.78m on KITTI-360.

Finally, motivated by the common ground between retrieval and pose estimation, we propose a novel view on CVGL that addresses the retrieval problem by representing it as an unsupervised pose estimation task. We integrate the method in a retrieve-and-rerank pipeline which significantly improves the recall of state-of-the-art methods, and is particularly effective in more challenging settings. Remarkably, the model learns to predict accurate camera poses *despite never seeing pose ground-truth during training*, and even achieves competitive performance with recent supervised approaches.

Zusammenfassung

Visuelle Geolokalisierung (VGL) beschreibt das Problem, den Aufnahmeort eines Fotos zu bestimmen, indem es mit einem georegistrierten Modell der Welt, z.B. einer Datenbank von Referenzbildern, abgeglichen wird. VGL stellt eine mögliche Alternative zu globalen Navigationssatellitensystemen (GNSS) dar, welche von der Verfügbarkeit eines externen Signals abhängig sind, und typischerweise nur eine Genauigkeit von einigen Metern erreichen.

In der Forschung werden seit langem Bilder aus einer Straßenansicht als Referenzdaten genutzt, z.B. solche, die über die Plattform Google Street-view bereitgestellt werden. Die spärliche Verfügbarkeit dieser Daten stellt jedoch eine erhebliche Einschränkung hinsichtlich der Skalierbarkeit und Kosteneffizienz dieses Ansatzes dar.

Luftbilder bilden eine mögliche Alternative für die Referenzdatenbank, gegen die Fotos aus einer Straßenansicht lokalisiert werden. Ihre weltweite und dichte Verfügbarkeit bietet das Potenzial, VGL auf viel größere Regionen zu skalieren und eine vollständigere Abdeckung zu erreichen als mit Fotos aus Straßenansicht in der Praxis möglich ist.

Die Aufgabe, Straßenansichtsbilder mit Luftbildern abzugleichen, stellt jedoch eine erhebliche Herausforderung dar, da entsprechende Methoden in der Lage sein müssen, den drastischen Perspektiv- und Maßstabsunterschied zwischen den Bildern zu überwinden. Das Problem wurde daher als Cross-view Geolokalisierung (CVGL) bezeichnet.

Bestehende Arbeiten in der Forschung nutzen einige einschränkende Annahmen, die zwar die Komplexität des Problems verringern, aber auch ihre Anwendbarkeit in realistischen Szenarios stark begrenzen. Darüber hinaus konzentrieren sie sich auf kleine Suchregionen in der Größenordnung einzelner Städte, für die die Geolokalisierung mittels Referenzdaten aus Straßenansicht bereits weit verbreitet ist und genutzt wird.

In dieser Arbeit betrachten wir die Aufgabenstellung der CVGL von Grund auf neu, und präsentieren neue Methoden, Datensätze und Erkenntnisse, die die Grenzen des Machbaren in Bezug auf Skalierbarkeit, Genauigkeit und Anwendbarkeit unter realistischen Szenarios erheblich vorantreiben. Wir gehen die Aufgabe durch eine Zerlegung in zwei Teilprobleme, Suche und Posen-Schätzung, wie folgt an.

Um Fotos in großen Suchbereichen zu lokalisieren, stellen wir eine neue Problemformulierung für die Suche vor, bei der das Gebiet in *gleichmäßig große* und *nicht überlappende* geographische Zellen unterteilt wird. Jede Zelle stellt eine Hypothese

für die Kameraposition dar und wird durch Luftbilder auf mehreren Auflösungsstufen repräsentiert. Ein Bild aus einer Straßenansicht wird dann lokalisiert, indem die zu ihm ähnlichste Zelle über ein Nearest-Neighbor Verfahren in einem gelernten Embedding-Raum bestimmt wird.

Unsere Arbeit ermöglicht es erstmals, Bilder aus einer Straßenansicht zu lokalisieren (1) in Suchregionen mit der Größenordnung ganzer Bundesstaaten wie Massachusetts mit 23000km^2, (2) unter realen Bedingungen mit handelsüblichen Kameras, (3) *in the wild*, d.h. ohne Informationen über Kameraeigenschaften wie die Brennweite, Linsenverzerrung oder Orientierung, und (4) ohne Zugriff auf Straßenansichtsbilder aus der Suchregion. So schafft es unsere Methode beispielsweise 60,6% aller nicht-panoramischen Fotos, die von Nutzern der Crowd-Sourcing Plattform Mapillary hochgeladen wurden, im Bundesstaat Massachusetts auf 50m Genauigkeit zu lokalisieren.

Um die genaue metrische Position und Orientierung einer Kamera zu finden, stellen wir ein neues, Ende-zu-Ende trainierbares Modell vor, das Fotos mit einem einzelnen, lokalen Luftbild abgleicht, um eine Wahrscheinlichkeitsverteilung über mögliche Posen auf dem Bild vorherzusagen. Wir führen ein zeitliches Filter ein, das die multimodalen Vorhersagen des Modells fortlaufend integriert, und so die langfristige Trajektorie einer Plattform schätzt.

Unsere Arbeit ermöglicht es erstmals, die geo-registrierte Ego-Pose und langfristige Ego-Trajektorie einer Plattform zu bestimmen (1) ausschließlich mit Luftbildern als Referenzdatenbank und ohne Zugriff auf Straßenansichtsbilder aus der Testregion, (2) unter Nutzung nur von visuellen Informationen und ohne Erforderlichkeit anderer Sensoren wie Lidar, Radar oder GNSS, und (3) mit einer Genauigkeit von unter einem Meter. Die Methode erreicht beispielsweise im Median einen Posen-Fehler von 0,87m auf dem Ford AV Datensatz, und im Durchschnitt einen Trajektorien-Fehler von 0,78m auf KITTI-360.

Motiviert durch die Gemeinsamkeiten unserer Methoden zur Suche und Posen-Schätzung, schlagen wir schließlich eine neue Perspektive auf CVGL vor, bei der das Suchproblem als eine unüberwachte Posen-Schätzungsaufgabe dargestellt wird. Wir integrieren diese Methode in einem Retrieve-and-Rerank Ansatz, der die Leistung von existierenden Methoden zum Suchproblem signifikant verbessert und sich besonders in anspruchsvolleren Umgebungen als effektiv erweist. Bemerkenswerterweise lernt das Modell genaue Kameraposen vorherzusagen, *obwohl es während des Trainings keine Posen-Grundwahrheit gesehen hat*, und erreicht sogar eine vergleichbare Leistung mit aktuellen überwachten Verfahren.

Acknowledgement

I would like to express my deepest gratitude to all *Wegbegleiter* - those who have accompanied and supported me throughout this journey. Without their help, encouragement, and patience, this dissertation would not have been possible.

First and foremost, I would like to thank my supervisor Rainer for his continuous support and guidance. Working as an external PhD student brings its own unique challenges, yet his mentorship, trust and expertise made all the difference. My sincere thanks also go to the members of my dissertation committee, and especially to Bastian Leibe for his insightful feedback.

Beginning a PhD feels like heading into an uncharted ocean - I was fortunate to be accompanied by an experienced navigator. Sebastian introduced me to the depths of academic publishing, stayed on deck during all crucial moments and helped balance the load of industry and academia. My PhD would not have been the same without his support. I am equally grateful to Christoph and Michael for many insightful conversations and providing the opportunity and freedom to explore new frontiers at Fraunhofer, and to my colleagues at OBJ and CV:HCI for fostering an inspiring and supportive research environment.

Lastly, I want to thank my family and Lena for their unconditional love and kindness. Thank you for always having my back.

Contents

List of Figures

List of Tables

List of Acronyms

API	Application Programming Interface	67
ATE	Absolute Trajectory Error	108
BEV	Bird’s Eye View	15
BPE	Byte Pair Encoding	58
C-BEV	Contrastive Bird’s Eye View Training	114
CLIP	Contrastive Language-Image Pretraining	30
CNN	Convolutional Neural Network	28
CPU	Central Processing Unit	93
CTRA	Constant Turn-rate and Acceleration	95
CVGL	Cross-view Geolocalization	11
DCL	Decoupled Contrastive Learning	59
DoF	Degrees of Freedom	14
EKF	Extended Kalman Filter	43
FoV	Field of View	11
GNSS	Global Navigation Satellite Systems	11
GPU	Graphics Processing Unit	30
HD	High Definition	85
HEM	Hard Example Mining	32
ICP	Iterative Closest Points	38
IMU	Inertial Measurement Unit	97
LoD	Level of Detail	53
LSE	log-sum-exp	118
MHA	Multi-head Attention	55
MLP	Multi-layer Perceptron	28
NIC	Navigation Integrity Category	12
NLP	Natural Language Processing	58
NMI	Normalized Mutual Information	39
OSM	OpenStreetMaps	38
PCA	Principal Component Analysis	106
PDF	Probability Density Function	111
PV	Perspective View	40
PV2BEV	Perspective View to Bird’s Eye View	116

RANSAC	Random Sample Consensus	115
RGB	Red-Green-Blue	13
RoI	Region of Interest	9
RTE	Relative Trajectory Error	108
SAFA	Spatial-aware Feature Aggregation	28
SfM	Structure-from-Motion	40
SLAM	Simultaneous Localization and Mapping	37
SMD	Spatial-mixed Feature Aggregation Module	28
SVGL	Street-view to Street-view Geolocalization	9
VGL	Visual Geolocalization	9
ViT	Vision Transformer	28
VL	Visual Localization	9
VLAD	Vector of Locally Aggregated Descriptors	28
VO	Visual Odometry	39
WMI	Weighted Mutual Information	38

Part I

Background

Introduction 1

1.1 Motivation

Humans determine their location in the environment primarily by relying on visual information which is matched against a mental model of the world. This has motivated a research problem coined Visual Localization (VL) which aims to estimate the location from which a photo is taken by relying only on its visual content. VL serves a wide range of applications from fields such as robotics, autonomous driving and augmented reality, and comes in many different flavors, for example to estimate relative motion in a video, or find the location from which a photo was taken by matching against a larger image database. When the desired reference frame in which a photo is localized is defined in a geographic coordinate system, the task is more precisely referred to as Visual Geolocalization (VGL) (*cf*. Fig. 1.1).

To perform VGL in a Region of Interest (RoI), a database of reference images is required that represents the model of the world against which a query image is localized. The reference images must be georegistered, provide coverage of all potential search locations and sufficient detail to allow for reliable and unambiguous predictions. Historically, this role has been filled by datasets of street-view images in the RoI that are captured manually or retrieved from online sources such as Flickr, Google Street-view or Mapillary [59, 121]. The corresponding task is referred to as Street-view to Street-view Geolocalization (SVGL)[1]. Large reference datasets ensure that for most query images at least one reference image exists that is captured from a nearby perspective; this provides comparable visual cues such that classical matching approaches are able to confidently find the correspondence and thereby predict the query's geolocation.

SVGL is however limited by the availability of street-view images in the RoI. While online platforms such as Mapillary provide a large quantity of data around the world, a sufficiently dense coverage of images with varying perspectives occurs mostly along main roads and landmarks, even in major cities (*cf*. Fig. 1.2). Developing regions often contain few to no images. Consequently, many state-of-the-art works focus on few regions with a sufficient density of street-view images, such as the city of San Francisco which is covered by the largest existing academic dataset for SVGL with an average of $\sim$28k reference images per km^2 [12].

[1] The abbreviation SVGL for this task is sometimes also defined as *single-view geolocalization* or *similar-view geolocalization*.

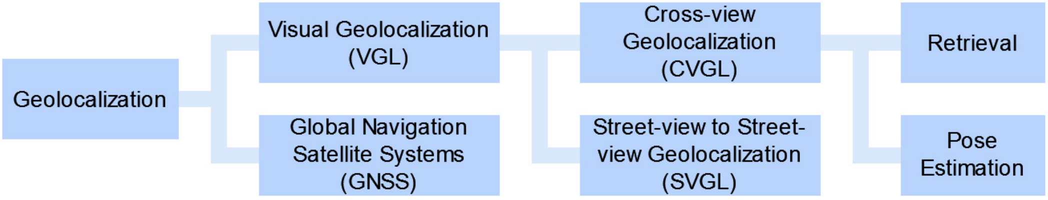

Geolocalization
Visual Geolocalization (VGL)
Global Navigation Satellite Systems (GNSS)
Cross-view Geolocalization (CVGL)
Street-view to Street-view Geolocalization (SVGL)
Retrieval
Pose Estimation

Fig. 1.1: Overview of geolocalization methods.

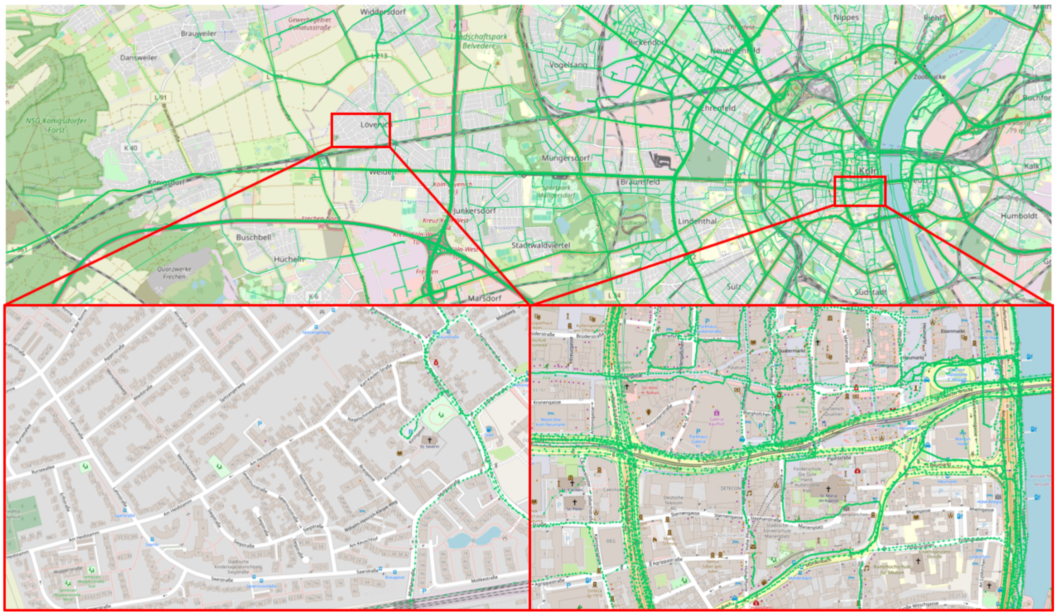


Fig. 1.2: Locations of street-view images available on the Mapillary platform in Cologne, Germany's fourth largest city. Images are captured mainly along roads and landmarks. The density is higher in the city center (right), but drops significantly outside of this area (left). The figure depicts screenshots from the Mapillary service.

To address the limited coverage and scalability of SVGL, a different line of research utilizes orthophotos as reference database against which street-view queries are localized. Orthophotos are captured from aerial platforms or satellites at large altitudes with a top-down perspective. They are orthorectified, *i.e.* geometrically corrected to provide a uniform scale by adjusting for the non-uniform ground elevation, camera distortion and perspective. Orthophotos are typically captured densely over an RoI and stitched together to form an orthophoto mosaic, *i.e.* a contiguous two-dimensional map of the region.

Using orthophoto mosaics as the reference database for VGL ensures a dense coverage of all parts of the RoI that are at least partially visible from an aerial perspective. Due to the wide availability in online databases such as Google Maps and Bing Maps and regional providers such as MassGIS [86] and OpenNRW [96], geo-registered orthophotos have the potential to serve as a world model for VGL that is both more complete and more scalable than existing sparse datasets of street-view images. They further allow extending to new regions with few existing street-view databases due to the fast and cost-effective nature of capturing new aerial imagery. For instance, high-resolution orthophotos for the state of Massachusetts with an area of $\sim$23000km^2 were collected in 14 flight days over a span of less than two months [86].

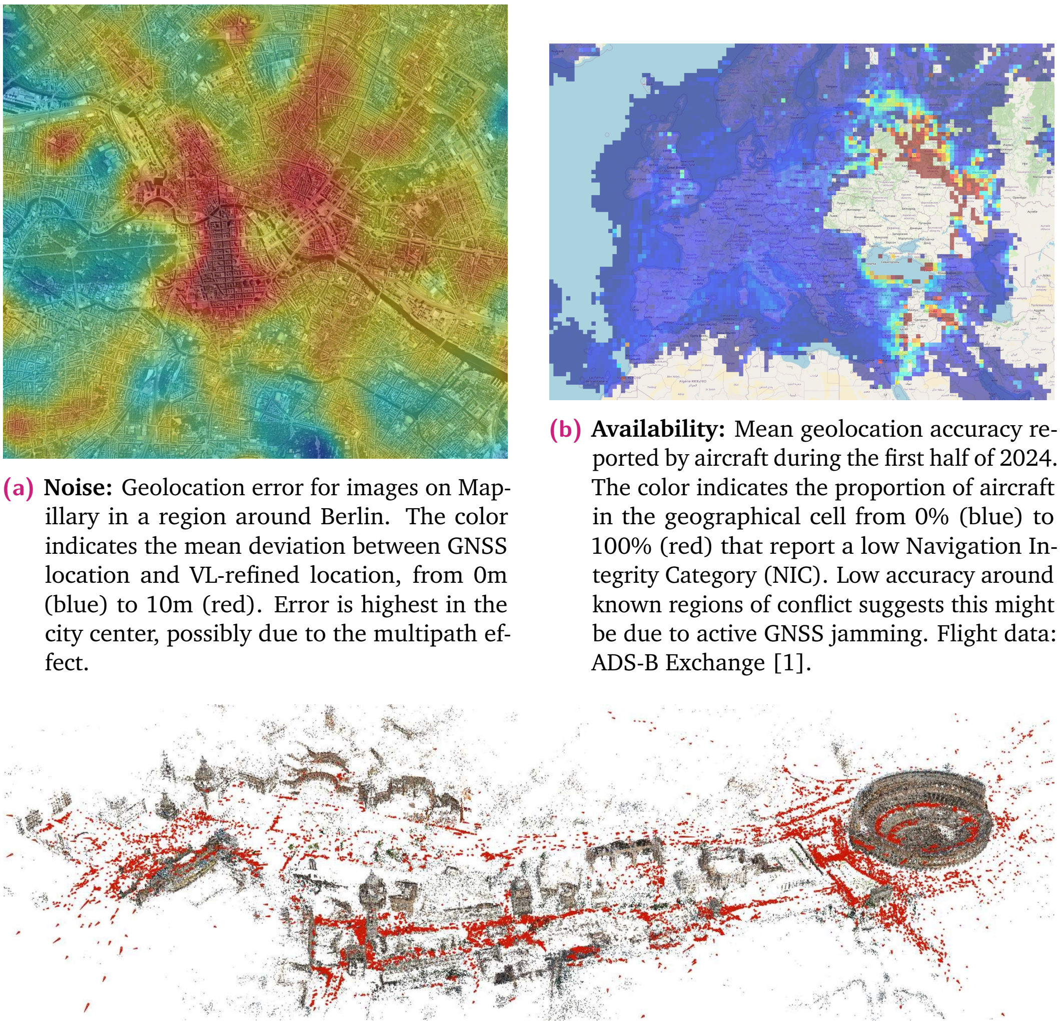
The matching between street-view queries and aerial-view references however poses a challenge to classical approaches due to the drastic changes in viewpoint and scale. The task has consequently been labeled Cross-view Geolocalization (CVGL). Recent works have shown the potential of modern deep learning methods to address this problem [107, 30], albeit relying on constraints that severely limit their real-world applicability and competitiveness with SVGL. For instance, widely used benchmarks for CVGL utilize north-aligned panoramas as query images that provide $360°$ Field of View (FoV) and require an additional compass reading [163, 144, 77]; in contrast, research in the field of SVGL focuses on consumer-grade pinhole cameras with limited FoV and arbitrary orientation [12]. CVGL in theory allows scaling to much larger regions than SVGL and operating without street-view imagery from the RoI. However, the largest existing benchmark in CVGL [163] covers a test region with a similar size as in SVGL [12], *i.e.* $\sim$100km^2, and many works rely on the availability of street-view data from the RoI during training to achieve good performance [107, 163].

To summarize, although the concept of using aerial imagery as reference data for VGL theoretically holds the potential to scale to large geographical regions under practical, real-world scenarios and without requiring corresponding street-view images, this capability has not been demonstrated in existing research.

Global Navigation Satellite Systems VGL stands in contrast to alternate methods that use Global Navigation Satellite Systems (GNSS) to provide a direct measurement of a sensor's geolocation. Mobile platforms such as vehicles and hand-held devices are typically equipped with GNSS receivers to allow for navigation in a georegistered reference frame. While GNSS provide a cheap and globally available method for geolocalization, they are subject to some limitations.

Firstly, GNSS typically do not achieve meter-level accuracy, especially with consumer-grade devices and in challenging environments such as urban canyons where the multipath effect [70] results in noisy measurements. For instance, Van Diggelen [123] report a mean accuracy of 4.9m for commonly used GNSS receivers. We find that GNSS locations of images uploaded to the Mapillary platform in a region around Berlin deviate from locations refined using VL by an average of $\sim$5m. This error increases to $\sim$10m in the city center (*cf*. Fig. 1.3a).

Secondly, GNSS require the availability of an external signal and are therefore potentially subject to adversarial attacks such as jamming, where the reception of signals is limited, and spoofing, where fake signals are emitted to fool platforms into



(a) **Noise:** Geolocation error for images on Mapillary in a region around Berlin. The color indicates the mean deviation between GNSS location and VL-refined location, from 0m (blue) to 10m (red). Error is highest in the city center, possibly due to the multipath effect.

(b) **Availability:** Mean geolocation accuracy reported by aircraft during the first half of 2024. The color indicates the proportion of aircraft in the geographical cell from 0% (blue) to 100% (red) that report a low Navigation Integrity Category (NIC). Low accuracy around known regions of conflict suggests this might be due to active GNSS jamming. Flight data: ADS-B Exchange [1].

(c) **Post hoc application:** The image shows a three-dimensional model of parts of Rome that is created from photos after they were uploaded to online platforms with little or no geographical information. GNSS cannot be used to provide geolocations in such scenarios, *i.e.* post hoc. Image created by Schonberger and Frahm [104].

Fig. 1.3: Problem cases surrounding the usage of GNSS.

estimating an incorrect geolocation. Reliable geolocations are particularly relevant in safety-critical applications such as autonomous driving. Researchers have for instance demonstrated the ability to spoof the GNSS sensor of a consumer-grade vehicle, causing it to follow unintended routes, adhere to an incorrect speed limit and initiate unwanted lane changes [90]. Fig. 1.3b shows geographical regions where active GNSS jamming was likely taking place during 2024.

Lastly, many applications utilize photos post hoc, *i.e.* after they are captured and uploaded to online platforms with little or no geographic information. For example, Agarwal *et al.* [2] use images downloaded from Flickr with the search terms "Rome" and "Roma" to build a three-dimensional model of parts of the city. Data from online sources without GNSS tags are available in large quantity and diversity, but require a purely vision-based method to be registered and used in a geographic reference frame.

1.2 Research Questions

Motivated by the above observations, the goal of this thesis is to develop methods for CVGL that push the existing boundaries of scalability and accuracy, and enable the usage in real-world applications with consumer-grade cameras. In this context, we address the following research questions.

RQ1: How to scale VGL to large geographical regions?

Finding the location of a street-view photo in a large RoI requires searching through and comparing with vast amounts of aerial imagery. The largest dataset that existing research in CVGL is directed at [163] however covers an area of just $\sim 115\text{km}^2$, *i.e.* 15GB of raw Red-Green-Blue (RGB) pixel data when using a pixel resolution of $0.15\frac{\text{m}}{\text{px}}$. It further focuses on major cities where the density of available street-view imagery tends to be high and SVGL approaches are therefore already widely adopted [12].

In contrast, a state-sized RoI such as Massachusetts is two orders of magnitude larger at $\sim 23000\text{km}^2$ and already corresponds to ~ 3.0TB of raw aerial imagery. On this scale, the features used for matching are required to be significantly more discriminative to be able to distinguish between different possible search locations. Additionally, the curvature of the earth's surface becomes a critical factor that has been overlooked in existing research. Lastly, methods cannot rely on memorizing the street-view scene layout from the test region during training due to the large size of the RoI and the much lower density of street-view images.

This thesis revisits the problem of CVGL from first principles, and proposes novel strategies for managing large RoI and for incorporating aerial imagery, a novel model architecture, and a training scheme which allows scaling to state-sized search regions.

RQ2: How to predict fine-grained camera poses with CVGL?

Since large-scale CVGL facilitates only rough location predictions up to tens of meters, a different line of works consider the problem of how to predict the exact pose with three Degrees of Freedom (DoF) w.r.t. a single aerial image that is centered on a given prior location. While research on large-scale CVGL has shifted to dedicated deep learning methods in recent years, pose estimation works still largely employ classical or hand-crafted feature matching approaches that are not able to robustly address the appearance change between street-view and aerial perspective without relying on additional input from range-scanners such as radar or lidar.

This thesis presents the first end-to-end learnable method for cross-view pose estimation that allows for application in a purely vision-based setup and is sufficiently accurate to enable long-term tracking of a platform's ego-trajectory with sub-meter accuracy.

RQ3: How to achieve robust CVGL under diverse, real-world conditions?

Real-world scenarios, such as a user capturing a photo with a hand-held smartphone, represent a significant challenge due to varying levels of image quality and noise, arbitrary camera orientations and potential lack of metadata such as the camera's intrinsic parameters or lens distortion. Existing approaches however heavily rely on the controlled conditions under which current benchmark datasets are captured; *e.g.* panorama images that provide a 360° FoV, are north-aligned via a known compass reading or taken with consistent camera orientations using a Google Street-view car. Furthermore, most methods for CVGL utilize street-view data from the test region during training, which limits their applicability to areas where such data is available in sufficient quantity and diversity.

This thesis presents methods that are robust and applicable to a diverse range of real-world scenarios across different geographical regions. We rely on a data-centric approach to address this problem by (1) utilizing new, large-scale datasets that are captured under diverse, real-world conditions, and (2) designing general models and training schemes that are able to exploit the diversity of training data and learn more general functions for CVGL.

1.3 Method

This section provides a short overview on our proposed methodology. We address the task of CVGL via a decomposition into two complementary sub-problems:

Retrieval considers large, up to state-sized search regions and provides a rough location estimate for the query photo with $\sim$50m accuracy. To do so, the search region is first partitioned into non-overlapping geographical cells. Our method aims to retrieve the cell from which the photo was captured using its orthophoto representation and thereby provide a estimate of its geolocation.

Pose estimation considers small search regions, *e.g.* with a radius of 50m around a prior location estimate, and aims to predict the exact camera pose within this region with three DoF, *i.e.* the longitudinal and latitudinal offsets, and a compass orientation. Towards this end, the street-view photo is transformed into a two-dimensional Bird's Eye View (BEV) representation of the environment and matched against a single, high-resolution orthophoto of the RoI.

The pose estimation allows refining prior locations that are determined for example from the retrieval step, a noisy GNSS measurement, or in a sequential setup where the pose is tracked continuously over a video.

1.4 Contributions and Outline

In the following, we provide a short outline of this thesis and an overview of our contributions towards addressing the problem of CVGL. They are grouped into three areas: A method for cross-view retrieval that allows localizing street-view photos in state-sized search regions in Chapter 4, a cross-view pose estimation approach that increases the fine-grained accuracy to a sub-meter level in Chapter 5, and a unified view on both sub-problems that facilitates among others the unsupervised learning of pose estimation in Chapter 6.

Cross-view Retrieval In Chapter 4, we revisit the first principles of cross-view retrieval and present a novel problem formulation, strategy for managing large RoI and for incorporating aerial imagery, a novel model architecture and training scheme. We introduce a new, large-scale retrieval dataset and provide experimental results. Our contributions for the first time allow localizing street-view photos

1. in state-sized search regions such as Massachusetts with 23000km^2 (**RQ1**),
2. without access to street-view data from the search region (**RQ3**),
3. under real-world conditions with consumer-grade devices (**RQ3**), and
4. in the wild, *i.e.* without requiring information about the camera's intrinsics, lens distortion or orientation during training or testing (**RQ3**).

In contrast, existing methods are able to address only small search regions up to $\sim 115\text{km}^2$ in size, and rely on controlled benchmark environments (*e.g.* panorama

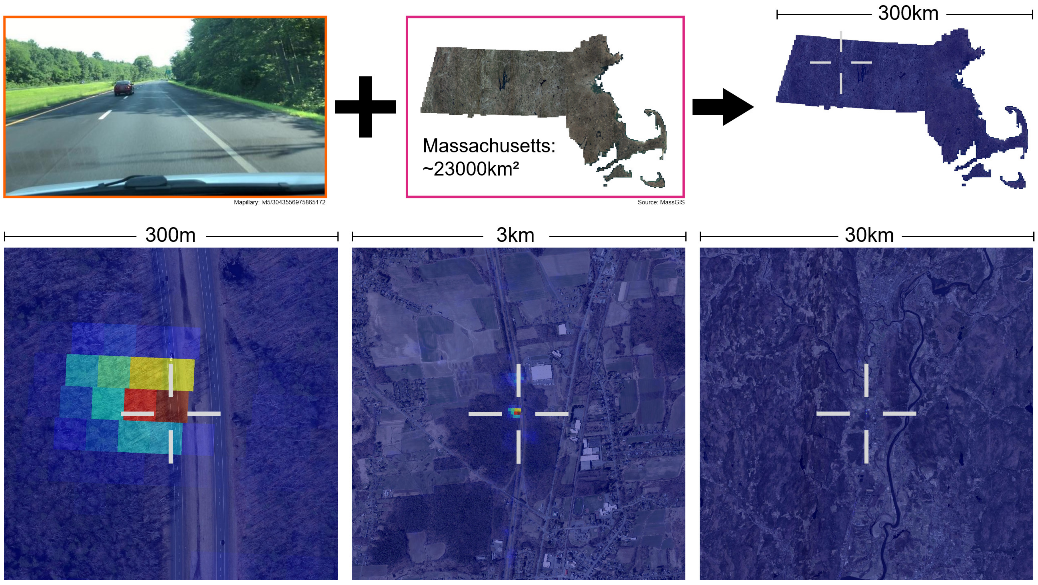


Fig. 1.4: Our **cross-view retrieval** method presented in Chapter 4 is able to localize street-view photos in the wild by matching against a database of aerial imagery in state-sized search regions. The method correctly localizes about 60% of all non-panoramic photos uploaded to the crowd-sourcing platform Mapillary in the state of Massachusetts to within 50m accuracy, without access to street-view data from this region during training and testing. The color indicates the predicted probability over possible locations in the search region from low (blue) to high (red).

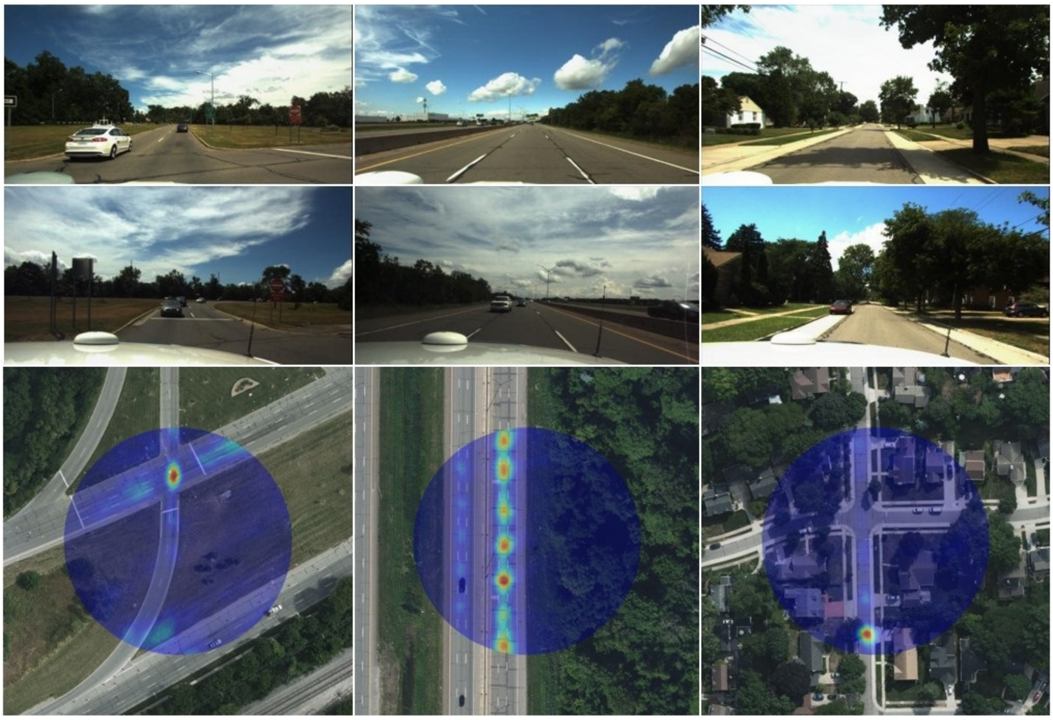


Fig. 1.5: Our **cross-view pose estimation** method presented in Chapter 5 is able to predict the geo-pose of a platform equipped with cameras to within sub-meter accuracy with three DoF. *Top rows:* The front and back camera of a vehicle. *Bottom row*: The predicted probability distribution over possible locations on an aerial image from low (blue) to high (red). The driving direction points upwards. Vehicle data: Ford AV dataset [3]. Map Data: Bing Maps © 2022 TomTom, © Vexcel Imaging.

images captured with a Google Street-view car and north-aligned via a known compass orientation) that prohibit application under real-world conditions.

Experimental results demonstrate that our method correctly localizes $\sim 60\%$ of all non-panoramic photos uploaded to the crowd-sourcing platform Mapillary in the state of Massachusetts to within 50m of their ground-truth location, without access to street-view data from this region during training and testing. Fig. 1.4 shows an example prediction.

This chapter is based on our publication in ECCV 2024 [37].

Cross-view Pose Estimation In Chapter 5, we present a novel end-to-end learnable model for cross-view pose estimation, as well as the respective loss formulation and training scheme. We propose a novel filtering framework that employs the predictions of the model to estimate a platform's trajectory over time. To evaluate the method, we introduce a new cross-view dataset based on several datasets from the field of autonomous driving, and refine the ground-truth using automatic pseudo-label and data pruning approaches. Our contributions for the first time allow determining the geo-registered ego-pose and long-term ego-trajectory of a platform

1. using only aerial imagery as reference database (**RQ2**) without access to street-view data from the test region or test vehicle (**RQ3**),
2. in a purely vision-based manner without requiring range scanners such as lidar or radar (**RQ2**) or external signals such as GNSS,
3. across a range of environments such as urban and rural regions (**RQ3**), and
4. with sub-meter accuracy (**RQ2**).

The method for instance achieves a median pose error and mean trajectory error of less than a meter on the Ford AV and KITTI-360 datasets, respectively. In contrast, existing vision-based approaches exhibit a median error of at least 5.0m on Ford AV despite being trained on street-view data from the test region. Our evaluation reveals that they rely mostly on prior information on the aerial image such as the location of roads and other drivable ground, rather than on information from the street-view photos.

Fig. 1.5 shows example predictions, and demo videos are available at:

`https://fferflo.github.io/projects/vismetcvgl23`

This chapter is based on our publications in CVPR 2023 [38] and IROS 2022 [36].

Unified View In Chapter 6, we present a unified view on cross-view retrieval and pose estimation. Motivated by the common ground between our work in Chapter 4 and Chapter 5, we propose a novel method that (1) utilizes pose estimation to address the retrieval problem following the retrieve-and-rerank paradigm and (2) is able to learn pose estimation in an unsupervised manner without requiring pose ground-truth during training.

We provide experimental results on several benchmark datasets. For instance, the recall of a state-of-the-art method [30] on the VIGOR cross-area split [163] with unknown camera orientation is improved from 31.1% to 65.0% when reranking with our method. Remarkably, the unsupervised pose estimation also achieves a localization accuracy that is competitive with recent supervised approaches.

This chapter is based on our publication in CoRR 2023 [35].

2 Basics and Related Works in Cross-view Retrieval

Cross-view retrieval aims to determine the geolocation of a street-view photo by retrieving the matching aerial image on which it is captured from a database of aerial images. Towards this end, the high-dimensional street-view and aerial images are mapped into a lower-dimensional embedding space (*cf*. Fig. 2.1) that captures relevant information and is invariant to distractors such as the scene's illumination or presence of dynamic objects. The distance between the embeddings reflects the predicted probability that the street-view image is captured in the geographical region represented by the aerial image.

This chapter gives an overview on the history of retrieval in CVGL, different problem formulations, presuppositions such as the translational and rotational alignment between street-view and aerial images, an overlooked scale inconsistency due to usage of a Mercator's projection, existing model architectures and loss functions, and a training scheme that relies on mining hard examples. The chapter concludes with a short summary of SVGL and comparison with retrieval-based CVGL.

2.1 History

Early research on retrieval-based CVGL before the widespread adoption of deep learning in the field of computer vision [65, 52] aims to address the embedding problem by designing hand-crafted descriptors for street-view and aerial-view images. For instance, Bansal *et al*. [9] find that classical visual descriptors such as SIFT [82] and MSER [87] that are designed for moderate viewpoint changes are not able to adequately address the appearance change in a cross-view setting. They instead construct a descriptor based on building facades that are extracted in both views using classical edge detection methods. Viswanathan *et al*. [126] address the deficiency of classical visual descriptors by first transforming the street-view photo into a BEV representation based on a flat-ground assumption. The BEV view more closely resembles the perspective of the reference imagery and allows building an embedding using classical descriptors such as SIFT [82], SURF [10], PHOW [15] and FREAK [4].

Workman *et al*. [144] and Lin *et al*. [75] present the first methods that use an embedding function that is trained to address the problem of CVGL. Workman *et al*. use an existing embedding model for the street-view images that is pretrained

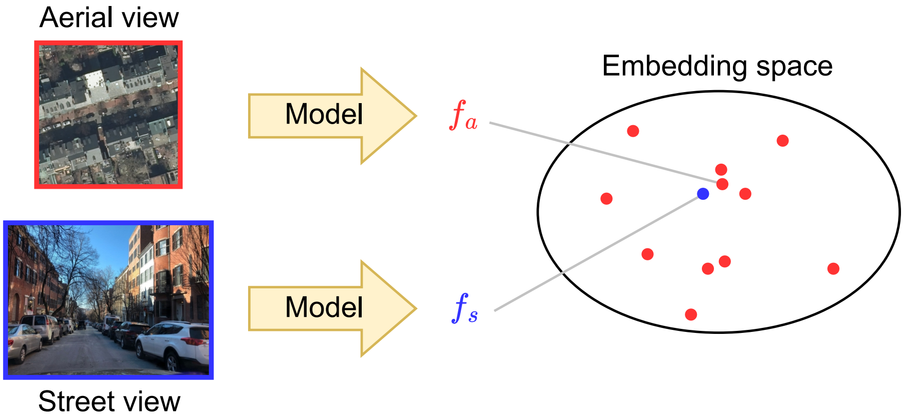


Fig. 2.1: Overview of the embedding function in the context of CVGL. Aerial-view and street-view images are mapped onto embedding features f_a and f_s. The distance between the embeddings reflects the predicted probability that the street-view image is captured in the geographical region represented by the aerial image. Given an ideal embedding function, a query image always matches the reference image that is closest in the embedding space.

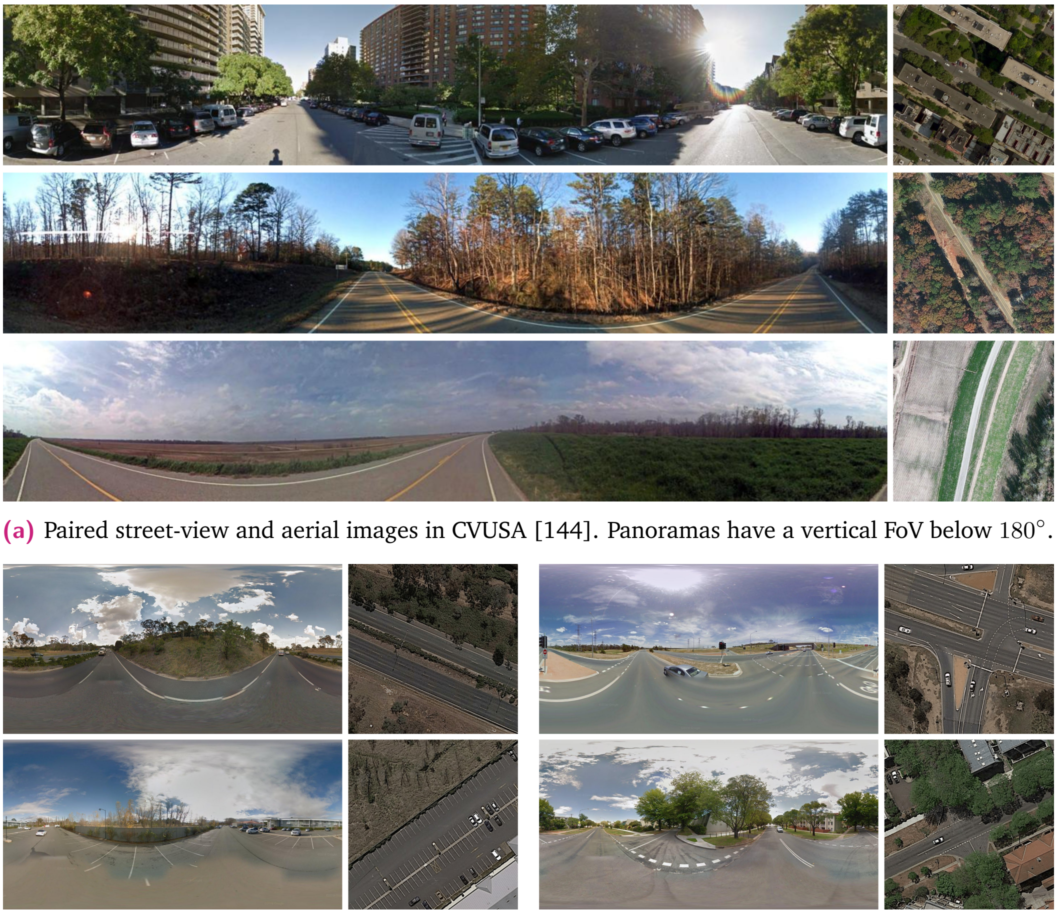
in an SVGL setting, and train only an embedding model for the aerial images. The loss function is formulated to align the aerial embeddings with the pretrained embeddings of matching street-view images.

Lin *et al.* [75] present the first work that trains models for both the street-view and aerial images *jointly*. The embedding space is therefore conditioned on both domains, rather than only on the street-view domain. They utilize a contrastive training objective [49] where the loss pulls the embeddings of matching street-view and aerial images together, and pushes non-matching embeddings apart.

Learning a joint embedding has become the defacto standard for retrieval in CVGL. The following sections discuss advances that have been made in this field since the work of Lin *et al.* [75].

2.2 Problem Formulation

The definition of the retrieval problem in published research on CVGL is heavily shaped by the benchmarks that methods are evaluated on. Several datasets have been released over the last decade that follow diverging problem formulations, albeit aiming at the same goal: To design methods with the ability to localize street-view photos in a RoI with as few constraints as possible. The datasets fit into one of two problem formulations: One-to-one matching and many-to-many matching.



(a) Paired street-view and aerial images in CVUSA [144]. Panoramas have a vertical FoV below 180°.

(b) Paired street-view and aerial images in CVACT [144]. Panoramas have a vertical FoV of 180°.

Fig. 2.2: Example of paired images in the CVUSA [144] and CVACT [144] datasets. Panoramas and aerial images are north-aligned: The center of the panorama depicts the upper region of the aerial image. The street-view camera is located at the center point of the aerial image.

2.2.1 One-to-one Formulation

The majority of existing benchmarks on retrieval-based CVGL represent a *one-to-one* matching task. The datasets are constructed by selecting a set of street-view photos for instance from Google Street-view, and sampling a single aerial image for each street-view photo's location. Each query therefore has exactly one matching reference image, and each reference image is matched to (at most) one query.

CVUSA [144] and CVACT [77] are similar datasets that contain a set of north-aligned street-view panoramas which are paired with aerial images centered on the respective camera locations. Fig. 2.2 shows corresponding examples of paired street-view and aerial images. The panoramas are shifted horizontally by the compass angle such that the image center always points north. Street-view cameras are located at the central points of the paired aerial images, up to GNSS errors. While CVACT provides panoramas with a larger vertical FoV, most works crop the additional pixels to make both datasets compatible.

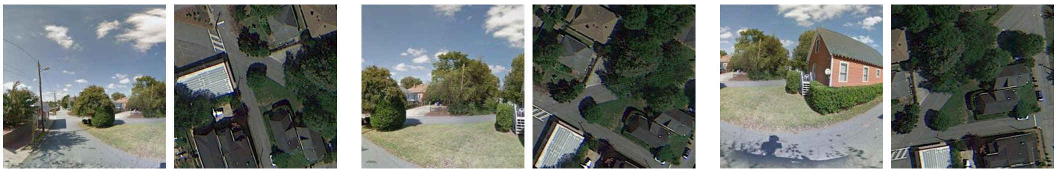


Fig. 2.3: Paired street-view and aerial images in the Vo&Hays dataset [130]. Street-view images are cropped from panoramas. Aerial images are aligned with the street-view orientation and centered on the depicted scene rather than the street-view camera location.

Vo and Hays [130] publish a one-to-one dataset that contains street-view images with limited FoV which are cropped from a set of panoramas. Fig. 2.3 shows corresponding examples of paired street-view and aerial images. The aerial images are rotated to align with the resulting camera orientation of the cropped images, rather than with the north direction. Unlike in CVUSA and CVACT, the center of the aerial image is positioned at some point in the depicted scene *in front of* the camera, rather than on the camera location itself.

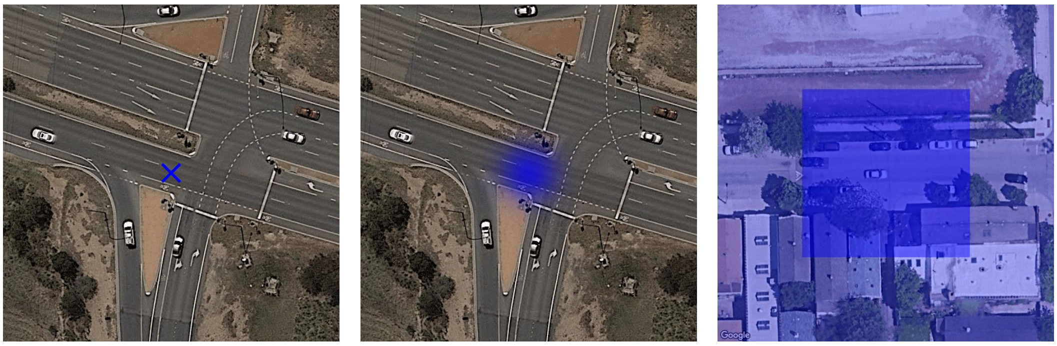
Zhang *et al*. [156] and Vyas *et al*. [132] aim to geolocalize street-view videos rather than individual photos and publish the SeqGeo and Gama datasets which contain a one-to-one mapping between video clips and aerial images. Each video matches with an aerial image that entails its trajectory. Consequently, the videos are chosen as small clips with a length of at most 50m [156] or 1 second [132].

The task represented by one-to-one matching benchmarks has received much attention in the research field. However, few works consider the question of how the setup can be extended to address the underlying goal of CVGL, *i.e*. to localize novel street-view images in a RoI. No existing one-to-one benchmark provides aerial imagery densely over a RoI that would allow testing this scenario.

We find that an extension of some of the benchmarks to the real-world scenario is not even possible in principle: The sampling of aerial images requires prior knowledge about the street-view locations that defeats the purpose of applying CVGL in the first place. *The locations of street-view images which are supposed to be geolocalized have to be known in advance to construct the aerial images that are utilized by the geolocalization method.*

For instance, each aerial image in CVUSA and CVACT is described to be centered on the camera location of the paired street-view image[1]. To localize a novel street-view image, an aerial image centered on its location must be contained in the reference database. Covering all locations in a RoI thus requires an infinite number of aerial images, since each aerial image corresponds only to a single geocoordinate (*cf*. Fig. 2.4a).

[1] Liu *et al*.: "without loss of generality, we assume the observer is standing at the center location of the satellite view" [77]



(a) Each aerial image in CVACT is defined to match to exactly one geocoordinate at the center of the image.

(b) Due to GNSS noise in the ground-truth positions, the aerial image in practice matches to a small, ill-defined region around the image center with decreasing probability for larger offsets.

(c) Each aerial image in VIGOR is defined to count as a *positive* match for camera locations in the central square region, and as *semi-positive* match for any other location on the image.

Fig. 2.4: Examples of different regions of street-view camera locations that an aerial image matches with in CVACT [77] (*i.e.* one-to-one setting) and VIGOR [163] (*i.e.* many-to-many setting). In a real-world scenario, the reference database of aerial images must cover all possible street-view camera locations in the RoI.

We find that noisy ground-truth locations result in small offsets between street-view and aerial images in CVUSA and CVACT, such that a model trained on the datasets also matches an aerial image against camera locations that are slightly shifted from the image center (*cf*. Fig. 2.4b). Vo and Hays [130] similarly choose the location of the aerial image w.r.t. to a noisy depth estimate in the street-view photo. However, it is unclear whether and how this allows efficiently covering an RoI with aerial images where each image corresponds to a non-zero area of possible street-view camera locations.

2.2.2 Many-to-many Formulation

Zhu *et al*. [163] publish the first and only existing dataset, VIGOR, that considers the question of how a RoI should be covered with aerial images. Given that a potentially infinite number of street-view camera locations must be matched against a limited number of aerial images, they propose a formulation where each reference image covers a well-defined, non-zero area of potential query locations. The aerial images further overlap, such that each query has multiple matching reference images. The setting thus represents a *many-to-many* matching task and allows localizing novel street-view photos in the RoI without prior knowledge about their positions. During training, the model is given matching query-reference pairs where the query photo is taken from within the matchable region on the aerial image, rather than only from the image center (*cf*. Fig. 2.4c).

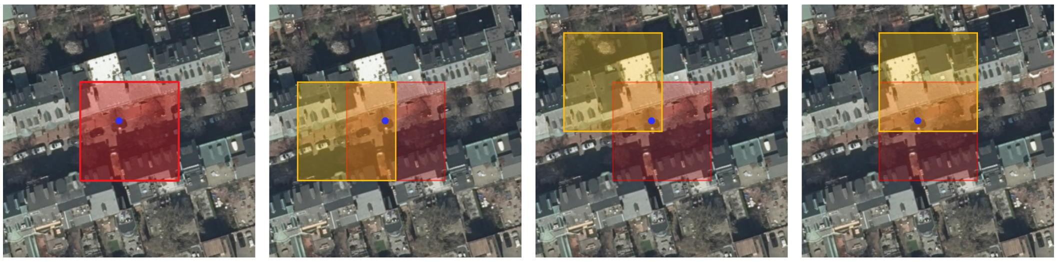


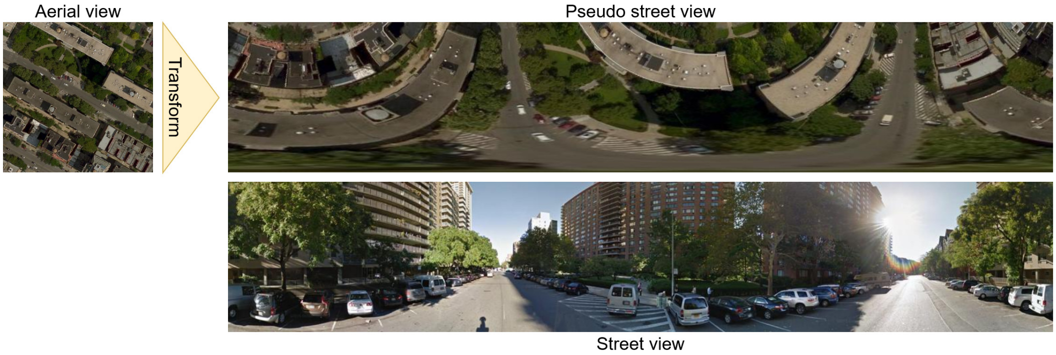
Fig. 2.5: Examples of correspondences between queries and references in a many-to-many matching setup as defined in the VIGOR dataset [163]. Any query location (blue dot) is covered by four reference images: One *positive* match (red square) and three *semi-positive* matches (yellow squares).

To construct the reference database, aerial images are chosen in a regular grid over the search region with a stride of half the image side-length, *i.e.* roughly 30m. Consequently, each potential query location is covered by four reference images. The nearest reference is defined as a *positive* match, and the remaining three references are defined as *semi-positive* matches (*cf*. Fig. 2.5). Each reference covers a non-zero area of potential camera locations that it is matched with. The task is thus characterized by *many-to-many* correspondences.

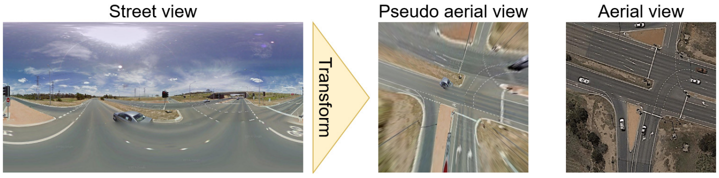
This strategy represents a *coupled* view of aerial images and search region cells: The extent of the image itself and the region of camera locations that it is matched with are identical. The matching across all locations on an aerial image is both supervised in the loss function via an explicit term for semi-positives, and encouraged via a novel metric called *hit rate*.

While the many-to-many formulation for the first time allows covering a RoI densely with aerial imagery, it is subject to some limitations. The size of the aerial image is an important hyper-parameter that defines the amount of context information available to the model to predict useful embeddings. However, increasing the size of the image also increases the region of potential query embeddings that have to be matched to the single reference embedding. This tends to have a negative impact on the resulting recall. A coupled view prohibits varying these parameters independently and finding the best choice for both. It further comes with the undesirable side-effect of defining multiple positive matches for each query if the aerial images are chosen to overlap in the RoI.

We revisit the problem of retrieval-based CVGL and propose a novel *many-to-one* formulation where each query is matched to exactly one search region cell, while using aerial imagery at a larger extent than the cell to predict the corresponding embedding. Furthermore, all datasets in the field of CVGL come with a predefined set of aerial images at the given resolution, scale and locations. This prohibits both (1) research into the choice of aerial imagery and (2) research into the layout of the search region into geographical cells. We address this problem by providing a



(a) Polar transformation of an aerial-view image from CVUSA [144] around the street-view camera's location into a pseudo street-view image as proposed by Shi *et al*. [108].



(b) Geometric transformation of a street-view image from CVACT [77] into a pseudo aerial-view image centered on its location based on a flat-ground assumption as proposed by Li *et al*. [72].

Fig. 2.6: Geometric transformations between street-view and aerial images.

general interface to retrieve arbitrary aerial images for the RoI and for the first time facilitating research in this direction.

2.3 Alignment Problems

Research on CVGL is shaped by the benchmarks that published methods are evaluated on. The benchmarks include assumptions about the input data that existing works rely on to address the corresponding task, but possibly limit their real-world applicability. In the context of retrieval-based CVGL this is observable among others in the use of a translation and rotation alignment between street-view and aerial images that exists in several benchmarks.

2.3.1 Translation Alignment

A street-view and aerial image are in translation alignment if their relative translation offset is zero, *i.e*. if the street-view camera is located on the aerial image's center. This is the case in widely used one-to-one matching benchmarks such as CVUSA [144] and CVACT [77] and limits their real-world applicability (*cf*. Sec. 2.2.1). Nevertheless, several works develop methods that explicitly rely on this alignment.

Shi *et al*. [108] propose computing a polar transformation of the aerial image around the street-view camera's location to bridge the domain gap between street view and aerial view (*cf*. Fig. 2.6a). They show that a model trained on the transformed images yields better results than training on the original aerial images. However, this requires the location of the street-view camera on the aerial image to be known in advance and therefore applies only to one-to-one matching tasks. Zhu *et al*. [160] show that the polar transformation degrades performance when applied in a real-world many-to-many setting (*i.e*. VIGOR) with translational offset between street-view and aerial images.

Li *et al*. [72] develop a method for training an embedding model in an unsupervised setting where the method is given a set of street-view and aerial images without knowing the ground-truth assignment between them. They address the issue by implementing a strong geometric bias in the training pipeline instead: The street-view images are transformed into an aerial perspective using a geometric transformation based on a flat-ground assumption (*cf*. Fig. 2.6b). The transformed images are refined using CycleGAN [158] and then used as the (pseudo-)matching reference images for the original street-view queries. The method yields reasonable results on CVUSA and CVACT, but fails to address VIGOR without at least portions of the corresponding ground-truth. The work attributes this to the "complex city scenes" [72] in VIGOR, but does not mention translational offsets that impact the generation of pseudo-matching aerial images for a given street-view image.

2.3.2 Rotation Alignment

All existing one-to-one and many-to-many benchmarks where the queries are given as panoramas with 360° FoV provide aerial and street-view images with rotation (*i.e*. azimuth) alignment. This requires the compass orientation of street-view camera to be known, which allows shifting the panorama horizontally such that the image center points north. Aerial images are typically given in a geo-registered reference frame and are north-aligned by default.

Training and evaluating a model on data that is rotation-aligned results in higher recall on retrieval benchmarks than training and evaluating with unknown relative rotation. For instance, the state-of-the-art method Sample4Geo [30] achieves a recall of 61.3% on the VIGOR cross-area split which is given with rotation alignment, but drops to 31.1% when simulating an unknown rotation via random shifting of the panoramas [35].

However, relying on a rotation alignment of the data limits scenarios that the method is applicable to. It requires the compass reading of the camera image to be known and thus prevents application in a post hoc scenario. Furthermore, shifting the street-view image to point north is only possible if it is given as a panorama with 360° FoV. Photos that are captured with consumer-grade cameras and limited FoV cannot be rotated analogously. In this case, the aerial images in the reference database must be augmented by a set of possible rotations, resulting in much larger reference databases and slower retrieval.

There is no consensus in the research community on whether datasets with north-aligned panoramas should be evaluated as-is, or by first undoing the rotation alignment via random horizontal shifts. The ambiguity has lead among others to research works making an unfair comparison between methods in an aligned and unaligned setting [162]. Some works propose methods that explicitly rely on rotation alignment, but limit their real-world applicability. For instance, Zhang *et al*. [155] rotate street-view and aerial images jointly for the purpose of data augmentation. Shi *et al*. [111] explicitly rely on the rotation alignment by projecting the street-view image into a north-aligned overhead view that is compared against the aerial image using a cross-correlation operation.

2.4 Scale Problem

CVGL aims to determine the geo-coordinates of a query image by utilizing aerial imagery of the RoI. The aerial imagery is given as a two-dimensional array of pixels, and cannot reflect an undistorted view of the earth's curved surface. Instead, a map projection has to be employed between planar pixel coordinates and spherical geo-coordinates that necessarily incurs distortions in the imagery.

The choice of map projection depends on the types of distortions that are acceptable in a given application. For instance, mapping service such as Google Maps, Bing Maps and Apple Maps use the Web Mercator projection [140] which is approximately conformal, but not area-preserving. Conformity ensures that angles between intersecting lines are preserved when mapping between planar coordinates and geo-coordiantes; images appear locally undistorted. However, any conformal map projection is not area-preserving [34]. In the Web Mercator projection, this results in the scale of aerial images changing based on their distance to the equator. For instance, if the aerial imagery is chosen with a pixel resolution of $1.0\frac{\mathrm{px}}{\mathrm{m}}$ at the equator, the resolution at a latitude of $\phi \in (-\pi, \pi)$ is increased to $\frac{1.0}{\cos\phi}\frac{\mathrm{px}}{\mathrm{m}}$; pixels cover less area and the image appears inflated when moving away from the equator.

All existing datasets in the field of CVGL [144, 130, 77, 163, 132, 156] sample aerial images from mapping services that employ a Web Mercator projection, and therefore implicitly incur area-distortions that change the scale of images based on their latitude. However, this scale inconsistency has so far been overlooked in the research community and has lead to undesirable consequences in several works.

For instance, the VIGOR dataset [163] includes incorrect ground-truth annotations [71] which we find to be due to the assumption that the pixel resolution is constant over different latitudes. Furthermore, the size of search region cells in VIGOR is coupled to the size of aerial images and thus varies based on the cell's latitude. As a consequence, regions away from the equator are covered with a higher density of cells and are therefore assigned a higher prior probability for matching with a query image. Lastly, training and testing in different regions at different latitudes represents a domain gap due to the varying image scales, which has lead to a significant, unexplained drop in performance when evaluating in a cross-dataset setting [152, 35].

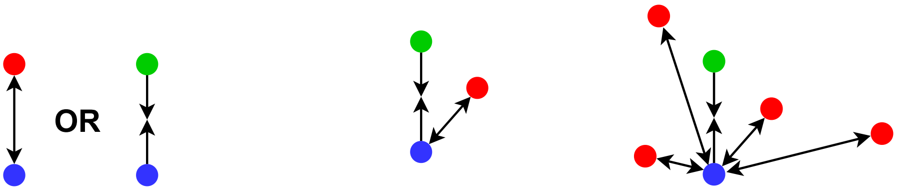
2.5 Model Architecture

Early works on CVGL use Convolutional Neural Network (CNN) architectures such as AlexNet [65] and VGG [23] to extract embeddings from the input images [144, 75, 130]. CNNs are widely used in computer vision and have shown impressive results in a variety of tasks such as image classification [29], object detection [76] and semantic segmentation [26]. However, they possess a limited receptive field and yield features that each capture information only about a local neighborhood in the input image.

To allow the model to reason about the global layout of the scene, several works design novel layers or reuse layers from other domains that provide a global receptive field over the image. Hu *et al.* [57] employ the NetVLAD layer [7] for CVGL that is designed to address a similar problem in the context of SVGL. NetVLAD is a trainable layer that is modelled after the classical Vector of Locally Aggregated Descriptors (VLAD). Shi *et al.* [108] and Zhu *et al.* [165] design similar layers specifically for CVGL called Spatial-aware Feature Aggregation (SAFA) and Spatial-mixed Feature Aggregation Module (SMD) which are based on a Multi-layer Perceptron (MLP) along the spatial dimensions.

NetVLAD, SAFA and SMD are designed to integrate a set of local features into a global image descriptor to address the limited receptive field of CNNs. Several works follow a different route by replacing (part of) the CNN with a Vision Transformer (ViT) [32, 124], *i.e.* a general purpose architecture for computer vision that intrinsically provides a global receptive field and is posed as an alternative to CNNs. Zhu *et al.* [160] employ a pure ViT model to embed input images in CVGL, while Yang *et al.* [152] and Zhang *et al.* [155] use a hybrid CNN and ViT model.

Recently, Deuser *et al.* [30] draw on a modernized CNN architecture, *i.e.* ConvNeXt [80], to outperform previous works and define the state-of-the-art in retrieval-based CVGL. They employ the CNN without a complex aggregation layer such as NetVLAD or SAFA, and simply mean-pool the set of local features to produce the global image embedding. In our work, we follow the choice of ConvNeXt as backbone, but design a novel aggregation layer that outperforms mean-pooling and other existing pooling layers.



(a) Pairwise contrastive loss. (b) Triplet loss. (c) InfoNCE loss.

Fig. 2.7: Comparison of different contrastive loss functions in the context of retrieval-based CVGL. The query (●) is contrasted with positive (●) and negative (●) references to train an embedding function. Positive samples are attracted (→←) while negative samples are repelled (←→) according to the definition of the loss function.

2.6 Loss Function

Lin *et al.* [75] present the first work that trains a joint embedding function for street-view and aerial images in CVGL. They utilize a *contrastive* training objective that pulls matching pairs of embeddings together and pushes non-matching embeddings apart. The corresponding loss function takes as input a street-view embedding f_s and aerial-view embedding f_a and is formulated as follows (*cf*. Fig. 2.7a):

$$\mathcal{L}(f_s, f_a) = \begin{cases} d(f_s, f_a) & \text{if } s \text{ and } a \text{ match} \\ \max(0, m - d(f_s, f_a)) & \text{if } s \text{ and } a \text{ do not match} \end{cases} \tag{2.1}$$

The term $d(f_s, f_a)$ denotes the (squared euclidean) distance between the embeddings, and m represents a margin parameter that sets the loss to zero for non-matching embeddings that are sufficiently far apart. The loss formulation is proposed by Hadsell *et al.* [49] in a general work that addresses a variety of different tasks.

The evolution of loss functions for CVGL in the following years is shaped by the advances of (contrastive) representation learning in other domains. Vo and Hays [130] present the first work that introduces the triplet loss [105] into the field of CVGL. Unlike the pairwise contrastive loss that is applied either to a matching or a non-matching pair, the triplet loss is applied to both a matching embedding f_{a^+} and a non-matching embedding f_{a^-} for a given query f_s (*cf*. Fig. 2.7b):

$$\begin{gathered} \mathcal{L}(f_s, f_{a^+}, f_{a^-}) = \max(0, m + d^+ - d^-) \\ \text{with } d^+ = d(f_s, f_{a^+}) \text{ and } d^- = d(f_s, f_{a^-}) \end{gathered} \tag{2.2}$$

This loss does not encourage matching query and reference embeddings to be exactly equal, but rather only requires the matching reference for a given query to be closer than (some) non-matching references. A soft variant of the triplet loss is proposed by Vo and Hays [130] and Hermans *et al.* [55] who replace the hard cutoff at margin m with a soft approximation using the softplus function with temperature τ:

$$\mathcal{L}(f_s, f_{a^+}, f_{a^-}) = \text{softplus}(\frac{1}{\tau}(d^+ - d^-)) = \ln(1 + e^{\frac{1}{\tau}(d^+ - d^-)}) \tag{2.3}$$

A disadvantage of the (soft or hard) triplet loss is its reliance on the availability of informative triplets: As the training progresses, the model tends to classify most randomly-sampled triplets correctly (*i.e.* $\mathcal{L} \approx 0$) and therefore receives little more supervision from these triplets. To address this problem, Ding *et al.* [31] propose computing the loss over *all* possible triplets in a given training batch, while Hermans *et al.* [55] find the *hardest* triplet constellation for each sample in the batch[2]. This ensures that for each query image the most informative reference images are used that are available in a given training batch, instead of sampling the reference images randomly.

Zhu *et al.* [165] introduce the InfoNCE loss [94] to the field of CVGL which was popularized by works on representation learning in other domains such as Contrastive Language-Image Pretraining (CLIP) [101]. For each query sample, InfoNCE considers one positive reference, and *all* negative references in the training batch with batch size b (*cf*. Fig. 2.7c):

$$\mathcal{L}(f_s, f_{a^+}, f_{a_1^-}, ..., f_{a_{b-1}^-}) = -\ln \frac{e^{-\frac{1}{\tau}d^+}}{e^{-\frac{1}{\tau}d^+} + \sum_i e^{-\frac{1}{\tau}d_i^-}} \tag{2.4}$$

InfoNCE is analogous to a formulation of the problem as a classification task: Each query must be assigned to one of b possible references that represent the set of classes in the batch. The loss is identical to the classical cross-entropy between the predicted distribution over all classes with temperature τ and the ground-truth (*i.e.* one-hot) distribution.

The classification-based perspective on retrieval on a per-batch basis suggests introducing other characteristics of classification-based loss functions to CVGL. For instance, Deuser *et al.* [30] apply label-smoothing [44] to the cross-entropy loss which reduces the stringent requirement for the correct class to be assigned a probability of 1 and serves a similar purpose as the margin parameter in the triplet loss.

[2] A training batch is a small set of street-view and aerial images for which the gradient w.r.t. the loss function is computed jointly during training and a single step of gradient descent is performed. The size of a batch is typically limited by the memory available on the training device, *e.g.* a Graphics Processing Unit (GPU).

Label-smoothing replaces the one-hot ground-truth distribution with a smoother distribution where the correct class is assigned $1 - \epsilon$ instead:

$$\mathcal{L}(f_s, f_{a_1}, ..., f_{a_b}) = \sum_i - p_i \ln \frac{e^{-\frac{1}{\tau}d_i}}{\sum_j e^{-\frac{1}{\tau}d_j}}$$
$$\text{with } p_i = \begin{cases} 1 - \epsilon & \text{if } s \text{ and } a_i \text{ match} \\ \frac{\epsilon}{b-1} & \text{if } s \text{ and } a_i \text{ do not match} \end{cases} \tag{2.5}$$

Eq. (2.4) represents a special case of Eq. (2.5) with $\epsilon = 0$.

Several works compute the triplet or InfoNCE losses in a symmetric fashion over queries and references [108, 160, 30]. In this setup, street-view images and aerial-view images are both used as queries and as references, and the final loss is computed as the mean of the two individual losses:

$$\mathcal{L}_{\text{sym}} = \frac{1}{2} \sum_s \mathcal{L}(f_s, f_{a_1}, ..., f_{a_b}) + \frac{1}{2} \sum_a \mathcal{L}(f_a, f_{s_1}, ..., f_{s_b}) \tag{2.6}$$

This formulation aligns with works from other domains such as CLIP [101] and has been shown to yield better retrieval results compared to training only with street-view images as queries.

Auxiliary Loss Some works suggest that a contrastive learning setup might not be sufficient for training accurate models for CVGL and propose including auxiliary objectives to aid the training process. Auxiliary objectives are represented by additional branches in the model and corresponding loss functions that supervise their output. The auxiliary model branches address a task that is distinct from the original (retrieval) task, but sufficiently related to improve the model's ability to address the original task. Auxiliary loss functions typically make use of additional ground-truth.

Several types of auxiliary loss functions have been proposed in the context of CVGL. Vo and Hays [130] and Cai *et al.* [19] add a model branch that regresses the relative heading angle between the street-view and aerial image and show that this yields better results than training only with a contrastive loss. While aerial imagery is typically north-aligned by default, using this loss function additionally requires the compass angle of street-view images during training. Zhu *et al.* [163] analogously add a model branch to regress the relative translation offset between the center of the aerial image and the position of the street-view camera on that image.

Toker *et al.* [119] follow a different route by adding a model branch that predicts a synthetic street-view image given the matching aerial-view image as input. During training, a generative-adversarial setup [45] is used to supervise the generation of aerial images. Regmi *et al.* [102] instead generate aerial images from street-view images.

Including auxiliary loss functions adds complexity to the training setup and requires additional ground-truth such as a compass angle or exact geolocations that are not always available. Furthermore, advances such as the InfoNCE loss in a CLIP-like training setup have significantly improved the results that are obtainable with a purely contrastive loss formulation. Recent state-of-the-art methods such as TransGeo [160], SAIG [165] and Sample4Geo [30] shift away from using auxiliary supervision and rely solely on a contrastive loss.

2.7 Hard Example Mining

During training of an embedding model in a contrastive setting, the model tends to quickly converge to a state where it classifies most randomly-drawn examples correctly (*i.e.* $d^+ < d^-$), and subsequently receives little more supervision. Hermans *et al.* [55] propose finding hard references for each sample in a training batch to address this problem, *i.e.* *in-batch* Hard Example Mining (HEM) (*cf*. Sec. 2.6). However, the problem of low supervision still arises a a later stage of training when all or most randomly-drawn examples per batch are classified correctly. Furthermore, recent loss functions such as InfoNCE [94] already leverage all negatives per batch and do not require additional in-batch HEM.

To find hard examples for each batch even during later stages of training, several works employ a *global* HEM strategy by searching for hard examples in the entire training dataset or a subset thereof. Each batch is then constructed from a set of hard examples and used in the training loop.

HEM requires the embeddings of all samples in the mining pool (*i.e.* the set of training images that is searched for hard examples) to be known. The embeddings are ideally predicted using the current model state at some point during training to find examples that are hard to classify for this specific model. Computing the embeddings for all samples in the mining pool after each step of gradient descent however is not computationally feasibly, especially if the size of the mining pool m is much larger than the batch size b. Two alternate strategies for determining embeddings for HEM have emerged in the field of VGL:

1. Predict the embeddings for the mining pool only every s iterations. The embeddings are compatible with each other due to resulting from the same model state, but are potentially outdated towards the end of the s iterations and result in easier batches and lower supervision. [7, 139, 30]

2. Keep the embeddings for all training samples in memory and update each embedding only when the respective sample is used in a training batch. This avoids the additional cost of inferring embeddings for the mining pool, but

results in embeddings that are potentially incompatible with each other due to originating from different model states. [162, 163, 160]

2.8 Street-view to Street-view Geolocalization

SVGL represents an alternate approach for VGL that utilizes street-view images rather than aerial images as the reference data. The following sections describe the two most common types of methods in this field, and analyze how they are related to retrieval-based CVGL.

2.8.1 Retrieval

The most widely used type of method for large-scale SVGL is based on a retrieval pipeline [5, 7, 6, 12, 13, 14, 51, 61, 150]:

1. Train an embedding function to map street-view images into an embedding space.
2. Construct a reference database of street-view images and store their embeddings in an efficient search structure.
3. For a given query at test time, find the reference image in the database that is closest in embedding space. The reference is assumed to be a match if it within some distance to the query's location (*e.g.* 25m [13]).

Similar to the many-to-many setup in CVGL, each query potentially matches to multiple reference images, and each reference covers a region of camera locations that it is matched with. Since the query and reference images are sampled from the same (*i.e.* street-view) domain, SVGL methods require only a single model as embedding function for both queries and references, rather than two separate models as in CVGL. Models are typically trained using contrastive loss functions such as the triplet loss, similar to CVGL.

2.8.2 Classification

A different line of research formulates the SVGL task as a classification problem. The search region is partitioned into geographical cells representing the set of classes, and a model is trained to predict the correct class for a given street-view query.

Most classification-based methods consider a partitioning of the entire earth into cells such that each cell covers a large geographical region. They predict the region from which an image is captured based mainly on image semantics rather than the specific scene layout in the immediate surroundings of the street-view photo. For instance, Vo *et al.* [129] and Seo *et al.* [106] use roughly 10^5 cells to cover the globe, resulting in a mean area of roughly 5000km^2 per cell, or 1500km^2 if covering only the globe's land area.

Recently, Trivigno *et al.* [122] apply a classification-based method at a finer resolution of 20m per cell over a smaller RoI of roughly 100km^2, and show that their approach yields better localization results than retrieval-based methods on large reference databases. However, they train on a dataset of street-view images from San Francisco that contains an average of more than 100 training photos per cell. A similar density of street-view images does not exist for most regions around the world.

At the finer resolution, classification-based SVGL resembles the many-to-many formulation of retrieval-based CVGL. In both cases, the RoI is partitioned into small geographical cells that represent the location hypotheses for a query photo. In retrieval-based CVGL, each cell is assigned an embedding vector *after* the training using aerial imagery from the RoI. Classification-based SVGL similarly represents each cell using an embedding vector. However, the embeddings are stored as part of the model weights and learned *during* training, as follows.

The last layer in a classification model maps the high-level features $f_S \in \mathbb{R}^c$ extracted from the street-view image in previous layers onto the relative scores for the n classes (*i.e.* cells) by multiplying with a weight matrix $W \in \mathbb{R}^{n \times c}$:

$$W \cdot f_S = \begin{bmatrix} — & W_1^T & — \\ — & W_2^T & — \\ & \vdots & \\ — & W_n^T & — \end{bmatrix} \cdot f = \begin{bmatrix} \ell_1 \\ \ell_2 \\ \vdots \\ \ell_n \end{bmatrix} \tag{2.7}$$

Each row W_i in the weight matrix corresponds to the c-dimensional embedding learned for the i-th cell in the RoI. The row is compared to the embedding vector f_S of the image via the dot-product similarity, resulting in a score ℓ_i for the cell.

In classification-based SVGL, a model learns embeddings for geographical regions (*i.e.* cells) using street-view images from these regions that it sees during training. The geographical regions are discrete and chosen in a grid-like manner over the RoI. To retrieve the embedding representation for a given point in the RoI, a simple lookup has to be done in the table of learned embeddings. A related work, GeoCLIP [128], replaces the lookup table with a learned function that directly maps geocoordinates onto embedding vectors. The function is trained similarly using street-view images from the RoI, and is used to test a set of potential geolocations for a given query image. Similar to most classification-based methods, GeoCLIP focuses on global application and has not been shown to address finer spatial resolutions, *e.g.* at 20m [122] or 30m [163].

Methods that learn representations for geographical regions during training, such as classification-based SVGL and GeoCLIP, require that street-view images from the RoI are available during training. The data must further be provided with a density that reflects the desired spatial resolution at which predictions are made. For instance, Trivigno *et al.* [122] achieve a spatial resolution of $20\text{m} \times 20\text{m}$ by training on one street-view image every $\sim$3m^2, on average. GeoCLIP achieves a lower spatial resolution of $\sim$200-1000km^2, and only trains on one image per $\sim 30\text{km}^2$ of global land area, on average[3]. In contrast, CVGL requires neither the availability of street-view images from the RoI, nor a sufficient spatial density of images during training.

[3] The work reports a recall of at least 50% at this resolution depending on the choice of the dataset.

3 Basics and Related Works in Cross-view Pose Estimation

Cross-view pose estimation assumes that a prior estimate of a street-view photo's location (and orientation) is given, *e.g.* up to 50m accuracy, and utilizes an aerial image centered on this location (and aligned with the orientation) to predict the exact pose of the camera with three DoF. Pose estimation and retrieval are complementary tasks: The retrieval typically predicts the location of a photo up to several tens of meters, and the pose estimation typically starts from a prior location estimate of several tens of meters. Other sources for defining the prior include a rough GNSS measurement or pose estimates from previous frames in a video.

This chapter gives an overview of research works on cross-view pose estimation, including the predominant paradigm of employing a Bird's Eye View (BEV) representation of the street-view data as well as non-BEV-based approaches. We summarize recent efforts to learn features for pose estimation in an end-to-end manner, and conclude with proposed extensions that also estimate a platform's trajectory over time.

3.1 BEV-based Methods

In the majority of works, explicit three-dimensional models of the environment are used to bridge the gap between the street view and aerial view. The models are determined for instance using range scanners (*e.g.* lidar, radar) or visual Simultaneous Localization and Mapping (SLAM) and allow projecting features from the street-view perspective into a BEV. The BEV represents an orthogonal, top-down view of the scene around the street-view camera that mirrors the aerial view, up to location and orientation alignment. It is given as a two-dimensional, sparse or dense map of features and allows matching with analogous features in the aerial view.

The following sections give an overview of methods that utilize the BEV to perform cross-view pose estimation. We distinguish methods based on the type of feature that is matched between the two views.

3.1.1 Vertical Structures

Vysotska and Stachniss [133] present a method that matches the measurements from a lidar scanner against an overhead map to determine its geo-pose. They extract building facades from the lidar point cloud using a line extraction algorithm and project into BEV by collapsing along the vertical axis. The BEV is matched via an Iterative Closest Points (ICP) approach against an overhead map that contains building outlines from the crowd-sourcing platform OpenStreetMaps (OSM).

Kim and Kim [62] extract a map of buildings from an aerial image using a semantic segmentation model instead, and directly match it against a lidar point cloud projected to BEV. The matching is performed by maximizing the Weighted Mutual Information (WMI) [48] between lidar points in BEV and building boundaries in the semantic map. The lidar points are not filtered to represent only buildings; instead, the method relies on the robustness of WMI against the large number of outliers that are not part of a building.

Two methods consider a more general class of vertical structures that is used as a matchable feature for the pose estimation. Kümmerle *et al*. [66] assume that the outline of vertical structures results in edges on the aerial image that are detectable with a classical Canny edge detector [20]. Similar structures are detected from the three-dimensional query point cloud (*e.g*. obtained from a range scanner or visual SLAM) as vertical edges with a large height variation. The features are projected to BEV and matched to estimate the pose.

Wang *et al*. [138] similarly use edges in the aerial image, but extract vertical structures from the query point cloud as follows. They divide the BEV into an array of bins and count the number of points per bin. Vertical structures in the point cloud result in large number of points in the respective bin, and a threshold over the bin count is used to define vertical edges.

Tang *et al*. [118, 117, 116] propose transforming the aerial image into a synthetic range scanner image that is analogous to a BEV of a point cloud measured by a real range scanner. Points in the point cloud that are at or below ground level are removed to focus only on vertical structures in the scene. The real and synthetic range scanner images are matched to determined the relative pose.

3.1.2 Horizontal Structures

Pink [100] and Javanmardi *et al*. [58] use lane markings as matchable features for the cross-view pose estimation. Pink extracts lane markings from the aerial images via a partially manual process, and from a street-view photo via a Canny edge detector. Since lane markings are assumed to be positioned on the ground plane, they use a simple homography to project the features into BEV. Javanmardi *et al*. extract lane markings via a thresholding technique over the aerial image and lidar intensity image.

Viswanathan *et al*. [127] consider a general ground-nonground distinction that is determined via a semantic segmentation of a lidar scan and projected to BEV via its three-dimensional information. A similar type of feature is extracted from the aerial image via a k-means clustering approach and matched against the query features.

3.1.3 Trajectory

Brubaker *et al*. [17] and Floros *et al*. [40] determine the geo-trajectory of a street-view video by utilizing a road map from OSM. A local trajectory for the video is first determined via a Visual Odometry (VO) approach that matches visual descriptors between subsequent frames to determine their relative transformation. The local trajectory is aligned with the road map to reduce the drift of the local pose estimation over time. The methods also address the place recognition problem over larger regions by finding paths that are compatible with the measured trajectory of the video. Similar to existing vision-based methods that determine a three-dimensional point cloud via visual SLAM without the use of range scanners, the methods of Brubaker *et al*. and Floros *et al*. also require a video of frames and are not applicable to single images.

3.1.4 Multi-class

Several works use a set of multiple types of features to perform cross-view pose estimation. For instance, Miller *et al*. [89] perform a full semantic segmentation of the aerial image and lidar scan to extract the road, terrain, vegetation and building classes. They determine the probability of a given pose by projecting the lidar points into the coordinate system of the aerial image and comparing the predicted classes between aerial pixels and projected points. Yan *et al*. [151] define a custom 4-bit descriptor based on the building and road classes extracted from aerial images and lidar scans.

3.1.5 Range Scanner Intensity

While most works on cross-view pose estimation use high-level features such as buildings or lane markings, Veronese *et al*. [125] and Vora *et al*. [131] propose directly matching the low-level lidar intensity values against the pixel values of the aerial image. The lidar points are projected into BEV and the corresponding intensity values are stored as a gray-scale image around the location of the lidar scanner. The aerial image is converted from RGB to gray-scale and directly matched against the lidar intensity image by measuring the Normalized Mutual Information (NMI) for different pose hypotheses.

3.1.6 Classical Visual Descriptors

Noda *et al*. [93] present a method that is based on the SURF descriptor which is typically used to match images from similar perspectives, such as in Structure-from-Motion (SfM) or SLAM. Due to the large change in perspective between street-view and aerial images, the street-view image is first transformed into BEV using a homography before extracting and matching SURF features to determine the relative pose.

3.1.7 Pose Regression

Zhu *et al*. [159] train a network to directly regress the relative pose between a lidar point cloud and an aerial image. The points are first transformed into a BEV map and concatenated with the aerial image along the channel axis. The data is then fed into the network which outputs three values for the three DoF of the relative pose and an additional score that predicts if the lidar scan and aerial image are captured from sufficiently nearby locations.

3.2 Perspective View to Bird's Eye View

The majority of cross-view pose estimation methods uses a BEV representation of query features which is matched against the aerial image to determine the relative pose. The BEV is particularly suited for the cross-view matching task, since it mirrors the aerial view, up to the relative location and orientation alignment. The problem of transforming features from Perspective View (PV) to BEV is called PV2BEV [83].

PV2BEV is trivial to solve if a range scanner (*e.g*. lidar, radar) is used to explicitly measure the three-dimensional structure of the environment; the height dimension is collapsed such that the two remaining axes match the dimensions of the aerial image. In the case of a purely vision-based system, visual SLAM allows estimating a three-dimensional model of the scene, but is more error-prone than a range scanner and requires a video with a sufficient amount of movement between frames. To address the PV2BEV task for single images in the context of cross-view pose estimation, several works employ a homography [100, 93, 107] that transforms features based on a flat-ground assumption. However, it fails to capture more complex types of scenes with objects of various heights.

Recently, more complex solutions to the single-image PV2BEV problem have gained attention in the research community in part due to their significance for autonomous driving tasks such as navigation, object detection and semantic segmentation in BEV. For instance, monocular depth estimation networks are trained to directly predict the depth of an input image as a discrete point cloud or probabilistic depth distribution, and allow transforming features from PV to BEV via the three-dimensional information, analogous to methods based on range scanners.

A different line of works propose dedicated layers for neural networks that transform features from PV to BEV. The layers are designed with a geometric bias that reflects the PV2BEV transformation, and are typically trained end-to-end without requiring additional geometric ground-truth such as depth-maps. For instance, Peng *et al*. [99] define a PV2BEV layer based on the transformer architecture [124] that is used to predict a semantic segmentation map in BEV from a given street-view image. The training requires only photos and matching segmentation maps as input; the segmentation loss supervises the network to learn the PV2BEV transformation as required.

While the PV2BEV problem has seen much innovation in the research community in recent years, methods other than a simple homography have not been leveraged in the context of CVGL.

3.3 Non-BEV-based Methods

Several works propose methods that estimate the pose in the cross-view setting without first transforming the query features into a BEV. Zhu *et al*. [163] train a model to perform retrieval and pose regression for street-view panorama images jointly, and attach the regression head to the embedding output instead of a BEV representation.

Xia *et al*. [145, 146] propose adapting cross-view retrieval methods to address the pose estimation task. In their first work, they sample aerial reference images at small relative distances such that the retrieval itself results in more accurate location estimate. They propose a modification of the triplet loss that enhances the discriminativeness of image embeddings in close proximity. In their second work, they predict a map of embeddings over the aerial image rather than a single embedding representation. Similar to the first work, the embeddings represent position hypotheses with smaller relative distances such that the local retrieval results in more accurate location estimates. They further define a network layer to refine the low-resolution probability distribution over the map of embeddings into a distribution with higher resolution.

Shi and Li [107] present an end-to-end trainable model based on a differentiable Levenberg-Marquardt optimizer that iteratively estimates the pose between a street-view and aerial image. The optimization is done by matching features in a street-view perspective rather than a BEV perspective.

3.4 Learning Features End-to-end

Most existing methods for cross-view pose estimation extract hand-crafted or predefined features from the street view and aerial view. This includes high-level features, such as buildings and lane markings [133, 62], and low-level features, such as edges in the image or range scan [100, 58].

High-level features are typically invariant to changes such as seasonal and daylight variations and are robust to the presence of dynamic objects like cars and pedestrians. However, they also discard a large portion of the information inherent in the input image that could potentially be leveraged for the pose estimation task. Furthermore, the methods require the presence of the particular type of high-level feature in the scene. For instance, the approaches presented by Vysotska and Stachniss [133] and Kim and Kim [62] utilize the building class and are not applicable to non-urban regions with a low density of man-made structures. The methods of Brubaker *et al*. [17] and Floros *et al*. [40] match the driven trajectory against a road map and are therefore not applicable to off-road scenarios.

Low-level features discard less information from the input images, but are also less robust to changes in the environment, *e.g*. due to daylight or seasonal variations or the presence of dynamic objects.

End-to-end trainable models have recently been introduced to the cross-view pose estimation task [118, 107] and offer the potential to learn features that are dedicated towards this problem, rather than designed by hand or reused from other domains such as semantic image segmentation. Learning features end-to-end allows the model to find a good trade-off between the invariance of existing high-level features and the discriminativeness of existing low-level features.

However, existing end-to-end trained methods for cross-view pose estimation are subject to some limitations. In the works by Tang *et al*. [118, 116, 117], the aerial image is mapped onto a synthetic range scanner image or occupancy grid that contains much less information than the input image and represents a bottleneck in the model's architecture. While this map is trained end-to-end, it is forced to discard large parts of the input information and thus cannot fully utilize the potential of the end-to-end paradigm. They further require range scanners rather than camera images and focus only on vertical structures by discarding points at or below the ground plane.

Shi and Li [107] use a flat-ground assumption to map features from aerial view to street view which limits the models ability to exploit features below or above the ground plane. Furthermore, the iterative Levenberg-Marquardt optimizer is prone to converge to local minima and does not explore the full space of pose hypotheses.

3.5 Trajectory Estimation

Many existing works integrate the pose estimation in a temporal filter to also estimate the trajectory over time. In this case, the prior pose at which the aerial image is sampled is defined using the predicted pose from previous frames in a video, and potentially includes a motion model that transfers the prediction to the current time step.

The most common approach is to employ particle filters, or sequential Monte-Carlo methods, that estimate the probability distribution over possible poses via a set of weighted samples [62, 66, 138, 127, 40, 151, 89] that are propagated over time and reweighted based on their accordance with the aerial image. Particle filters are particularly suited for noisy pose estimations due to their ability to capture multi-modal distributions.

Vysotska and Stachniss [133] and Zhu *et al*. [159] adapt a SLAM system to integrate the predicted geo-poses. SLAM estimates the local trajectory of the platform and is subject to an increasing drift over time. The predicted geo-poses are included in the pose graph to provide constraints w.r.t. a geo-registered coordinate system, such that the drift is reduced to within the registration error. All constraints are optimized jointly using a least-squares approach.

Vora *et al*. [131] present a method for trajectory estimation based on an Extended Kalman Filter (EKF). The EKF tracks a uni-modal distribution over poses that allows predicting the trajectory with high accuracy, but is less robust to noisy pose measurements than the particle filter.

Part II

Visual Cross-view Geolocalization

Revisiting Retrieval for Cross-view Geolocalization 4

4.1 Motivation

Finding the location of a query photo in a large RoI requires searching through vast amounts of aerial image data to determine if a given location represents a match. For example, a state-size search region such as Massachusetts that covers a land area of $\sim 23000 \mathrm{km}^2$ and provides aerial imagery at $15 \frac{\mathrm{cm}}{\mathrm{px}}$ corresponds with a database of 3.0TB of raw image data. Directly processing this data at test time to compare it against a query photo would result in long search times; roughly one day when using the popular machine learning model ConvNeXt [80] in its base configuration on an A100 GPU.

To search through the large amount of reference data, we instead employ a retrieval-based approach based on the following steps:

1. Partition the search region into non-overlapping, geographical cells with a size of 30m×30m. Compute an embedding representation for each cell that captures relevant characteristics in the corresponding aerial imagery and allows matching it against a street-view photo (*cf*. Fig. 2.1). The embeddings are determined *before* deployment, such that the processing cost occurs only once and amortizes over many queries.
2. At test time, map a given query onto its embedding representation and find the geographical cell in the reference database with the most similar embedding. The similarity between a query and reference sample is determined via a simple distance metric in the joint embedding space (*e.g.* euclidean distance or cosine similarity). This allows employing approximate nearest neighbor algorithms with sub-linear computational complexity to significantly reduce the search time per query in large databases.

Mapping query and reference images into a joint embedding space that allows for effective retrieval represents a significant challenge due to their vastly differing appearance. Street-view photos are captured from within the scene with varying detail of objects based on their distance to the camera, while aerial-view photos are captured at large distances to the scene with an orthographic projection. The images are further subject to different sources of image noise, such as atmospheric conditions in the aerial view, and defocus or motion blur in the street view that occurs in real-world application with consumer-grade devices. The query image is generally not captured synchronously with the reference imagery, requiring the mapping to generalize over long-term temporal changes in the scene:

- Dynamic objects such as vehicles and pedestrians do not appear in the same locations in both images.
- The illumination of the scene in the street-view images changes based on daylight and other factors, while aerial imagery is typically captured with optimal lighting conditions during the day.
- The appearance of the scene changes due to seasonal variations, such as the vegetation's leaf coverage, and weather variations, such as rain and snow. Aerial imagery is ideally captured with a low cloud coverage and without snow.

Classical approaches [9, 126] that employ embedding functions based on hand-crafted descriptors (*e.g.* SIFT [82], SURF [10]) have not been demonstrated to adequately address the discrepancy between street-view and aerial-view images on large-scale benchmarks. Research on CVGL over the last decade has instead shifted to using embedding functions that are learned in a data-centric manner [160, 30, 108]. Several works have demonstrated the potential of deep learning models to address the large appearance changes between street view and aerial view. However, they rely on constraints that severely restrict their real-world applicability and have lead to limited adoption of CVGL outside of an academic context.

For instance, most methods rely on the usage of panoramas that provide 360° FoV and are easier to match than more widely used consumer-grade photos with limited FoV, albeit being much less ubiquitous; or they rely on a rotation alignment between street-view and aerial images that requires an additional compass sensor and mainly addresses the use-case of panoramas (*cf*. Sec. 2.3.2).

Furthermore, the largest dataset in the field of CVGL, *i.e.* VIGOR [163], is similar in size to existing datasets used in SVGL such as SF-XL [12] at roughly 100km^2, and focuses on major cities where the density of available street-view imagery tends to be high and SVGL is therefore already widely adopted. While one of the advantages of using aerial imagery as reference data is its scalability to large search regions without access to corresponding street-view images, this capability has not been demonstrated in existing works.

Lastly, there are several limiting factors that have been overlooked in the research field, such as the non-transferability of one-to-one matching to a densely covered RoI (*cf*. Sec. 2.2.1), and the inconsistent scale of aerial imagery due to map projections that are used implicitly in map providers such as Google Maps (*cf*. Sec. 2.4) and severly impact the performance of CVGL models.

4.2 Overview

In this chapter, we present our research on retrieval-based CVGL. We propose a novel problem formulation that addresses the limitations and ambiguities in existing benchmarks, and present a comprehensive method to address this task and achieve robust geolocalization in much larger RoI and under fewer constraints than previously possible. We introduce a novel, large-scale and diverse dataset that allows training and evaluating methods in more than $100\times$ larger RoI than previously possible. Our contributions for the first time allow localizing street-view photos

1. in state-sized search regions such as Massachusetts with 23000km^2 (**RQ1**),
2. without access to street-view data from the search region (**RQ3**),
3. under real-world conditions with consumer-grade devices (**RQ3**), and
4. in the wild, *i.e.* without requiring information about the camera's intrinsics, lens distortion or orientation during training or testing (**RQ3**).

We describe our revised problem formulation which is based on a *many-to-one* matching task in Sec. 4.3.1, and address the details of our proposed approach w.r.t. the different components of retrieval-based CVGL in the following sections, including the choice of aerial imagery in Sec. 4.3.2, our proposed model and loss function in Sec. 4.3.3 and Sec. 4.3.4, a novel HEM strategy in Sec. 4.3.5 and our data augmentation scheme in Sec. 4.3.6. We give an overview of our large-scale dataset in Sec. 4.4 and provide the results of our evaluation and ablations studies in Sec. 4.5.

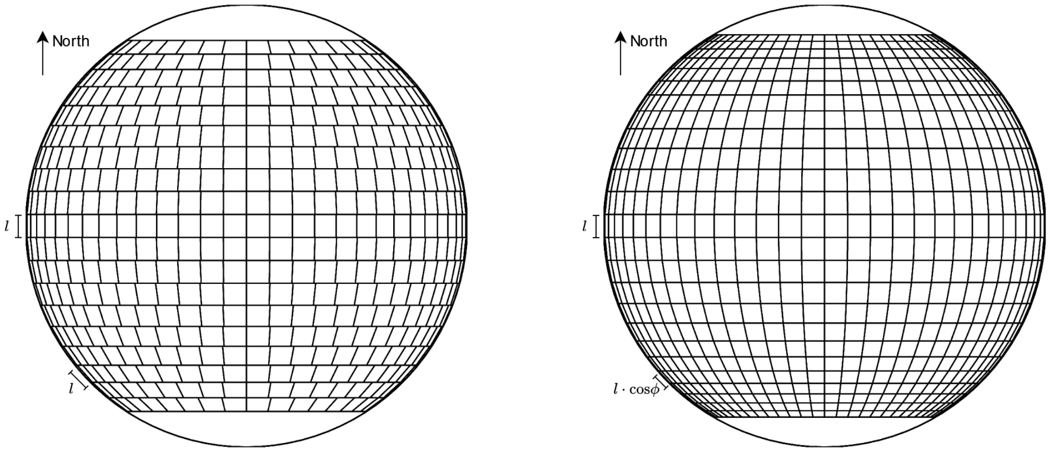
This chapter is based on our publication in ECCV 2024 [37].

4.3 Method

4.3.1 Problem Formulation

The underlying goal of retrieval-based CVGL is the following: Given a query image that is captured somewhere in a RoI, find its location by matching against a dense coverage of aerial imagery. We revisit this task and present a novel strategy based on *many-to-one* correspondences that avoids the drawbacks of the existing one-to-one (*cf*. Sec. 2.2.1) and many-to-many formulations (*cf*. Sec. 2.2.2).

We partition the RoI into geographical cells that are *non-overlapping* and *densely sampled*. Each potential camera location thus matches with exactly one cell (*i.e.* one reference per query), and each cell covers an area of multiple potential camera locations (*i.e.* many queries per reference). In contrast, one-to-one matching defines



(a) Our consistent search region layout.

(b) Regular grid over a Mercator projection. Size of cells changes with the latitude ϕ.

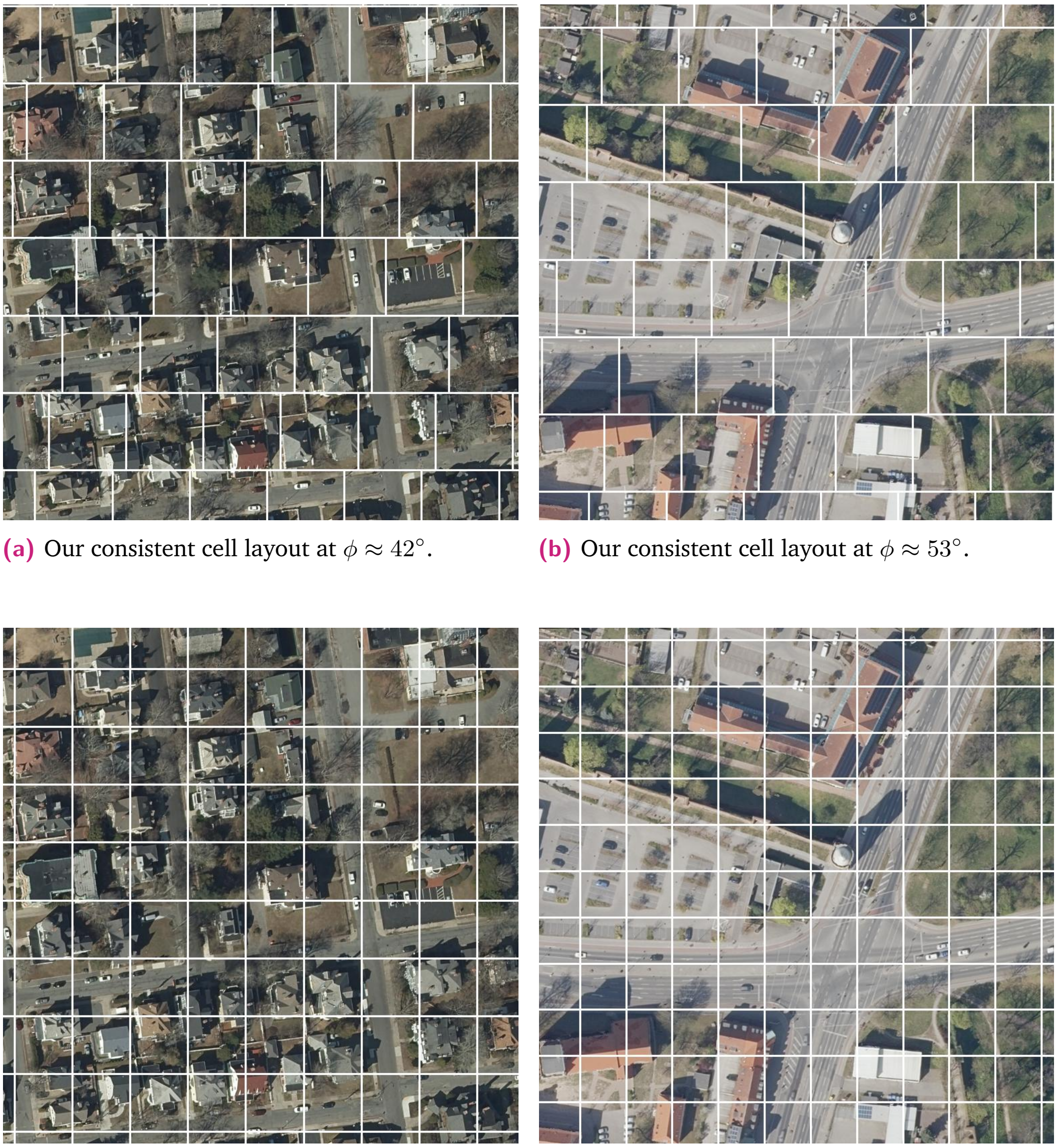
Fig. 4.1: Overview of different strategies for the partitioning of the globe into geographical cells. The images depict an orthographic projection of the globe. For better illustration, cells are shown with a large side-length of $l = 6°$ at the equator, rather than with $l = 30$m. Regions near the poles are not covered.

only one matching query location per reference (*cf*. Sec. 2.2.1), and many-to-many matching defines many positive references for each query location (*cf*. Sec. 2.2.2).

We predict an embedding representation for each cell from aerial imagery that provides a view of the scene around the cell. At test time, the embedding of a street-view image is used to query the reference database and retrieve the matching geographical cell. We choose the shape and size of all cells and corresponding aerial images to be consistent and independent of their geographical location. This has the following advantages:

- A model trained on data from one region is also applicable to data from other regions. Existing datasets in the field of CVGL use a Mercator projection which instead changes the metric size of cells and aerial images based on the image's latitude (*cf*. Sec. 2.4). This represents a domain gap which prevents scaling the methods to large geographical regions.
- All regions around the globe are assigned the same prior probability of being matched to a query due to the consistent density of cells. In contrast, the density of cells in the VIGOR dataset [163] increases with larger distances to the equator due to the overlooked scaling effect of the Mercator projection.

In the following, we address the question of how the globe is partitioned into equally sized and shaped cells. A division according to a regular grid is not possible due to the curvature of the earth's surface. In VIGOR, a grid is instead applied to a flattened map projection of the globe, resulting in consistent cell sizes in projected coordinates, but not in sphere coordinates (*cf*. Fig. 4.1b).



(a) Our consistent cell layout at $\phi \approx 42°$.

(b) Our consistent cell layout at $\phi \approx 53°$.

(c) Mercator-based cell layout at $\phi \approx 42°$ with a scale factor of $\cos\phi \approx 0.74$.

(d) Mercator-based cell layout at $\phi \approx 53°$ with a scale factor of $\cos\phi \approx 0.60$.

Fig. 4.2: Examples of search region layouts at the latitudes of $\phi \approx 42°$ in the US and $\phi \approx 53°$ in Germany. The size of cells in our layout is consistent and independent of the geographical region. The size of cells in a Mercator-based layout is inconsistent and reduced by a factor of $\cos\phi$ w.r.t. the equator. Cells are chosen with a side-length of 30m at the equator in all examples.

We propose a two-step process to partition the globe into cells of size $l \times l$ (*cf*. Fig. 4.1a). We assume a spherical model of the earth with radius r.

1. Partition the globe into longitudinal rows with a height of l meters, *i.e.* $\frac{l}{r}$ radians. The i-th row north or south of the equator is centered on a latitude of $\phi_i = i\frac{l}{r}$ and has a radius of $r_i = r\cos\phi$ that is reduced by a factor of $\cos\phi_i$ w.r.t. the equator.
2. Partition each row independently into cells with a width of l meters. Since the circumference of the i-th row around the globe is reduced by a factor of $\cos\phi_i$ w.r.t. the equator, the angle per cell is increased to $\frac{l}{r_i}$ radians, and there are fewer total cells in the row than at the equator.

Fig. 4.2 shows examples of search region cells at different latitudes in the US and Germany. Cells in our consistent layout have the same size independent of their geographical location. Rows of cells appear shifted w.r.t. each other due to their varying circumference and number of cells. In a Mercator-based cell layout [163], the size of cells at a latitude of ϕ is reduced by a factor of $\cos\phi$, resulting in a higher prior probability for the region and a domain gap when training and testing at different latitudes.

In both layouts, the shape of cells deviates slightly from a perfect square due to distortions from the earth's curvature. For instance, in the Mercator-based layout, vertical lines appear to be parallel in the flattened coordinates, but intersect at the poles (*cf*. Fig. 4.2). However, at the scale of individual cells with $l \ll r$ this effect is negligible. In the following, we estimate the approximation error for our consistent cell layout.

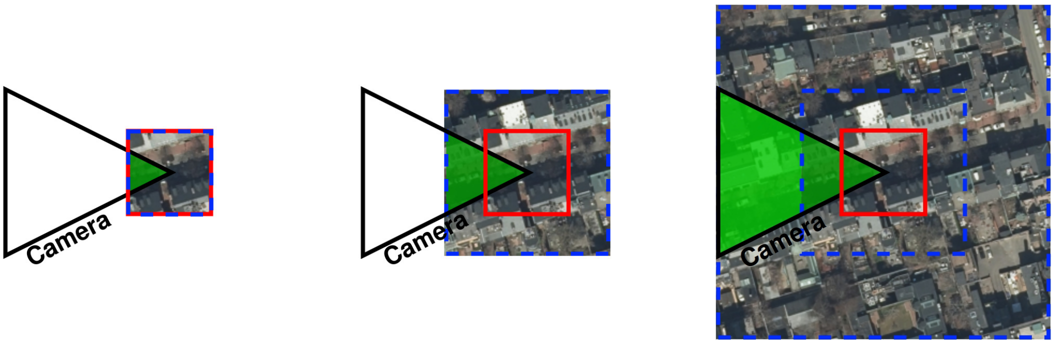
Consider a cell in the northern hemisphere that is positioned in a longitudinal row between the latitudes ϕ_1 and $\phi_2 = \phi_1 + \frac{l}{r}$ with $\phi_1 < \phi_2$. The longitudinal circumference of the row at the lower and upper ends is defined as

$$c_{\{1|2\}} = 2\pi r \cos\phi_{\{1|2\}} \ . \tag{4.1}$$

Given that $c_1 > c_2$, the cells have a trapezoid shape with a ratio between the shorter and longer sides of roughly

$$k = \frac{c_2}{c_1} = \frac{\cos\phi_2}{\cos\phi_1} \ . \tag{4.2}$$

At the equator, $k \approx 1$ and the cells are roughly square-shaped. At larger latitudes ϕ, the ratio k decreases and the cells become more triangle-shaped. However, when using a cell size of $l = 30$m and latitudes in the range $\phi \in [-85.06°, 85.06°]$ (*i.e.* the boundaries of mapping services such as Google Maps [140]), the ratio k is bounded by $k > 1 - \epsilon$ with $\epsilon \approx 6.3 \cdot 10^{-4}$. This corresponds to a difference between the upper and lower sides of the cell of at most 1.9cm and is much smaller than the resolution at which aerial imagery is typically provided, *e.g.* $15\frac{\text{cm}}{\text{px}}$ in Massachusetts [86].



(a) Strategy 1: Single aerial image coupled with the search cell.

(b) Strategy 2: Single aerial image decoupled from the search cell.

(c) Strategy 3: Multiple aerial images decoupled from the search cell.

Fig. 4.3: Three strategies for choosing aerial images for a search cell to predict its embedding. The solid red square depicts the search cell. A dashed blue square delineates a single aerial image. The green triangle shows the overlap between camera frustrum and aerial images. Larger overlap provides more information to the model from which matching embeddings are predicted.

4.3.2 Choice of Aerial Images per Cell

Given a search region that is partitioned into geographical cells, we assign an embedding representation to each cell that allows matching it against a street-view photo to determine if the photo is captured from within the cell. We consider three different strategies for sampling aerial images for a cell from which its embedding is computed (*cf*. Fig. 4.3).

The first strategy (*cf*. Fig. 4.3a) uses a single aerial image that is coupled to the size of the corresponding search cell. While choosing large images and cells potentially gives a large overlap with a corresponding street-view photo, this is not always the case: Photos that are captured close to the cell boundary and face away from the cell center always have a low overlap and are therefore difficult to match. Due to the many-to-one formulation, no additional reference images are included that would provide better coverage for such locations.

The second strategy (*cf*. Fig. 4.3b) uses a single aerial image that is decoupled from and larger than the corresponding search cell. This results in more overlap with the street-view photo than in Fig. 4.3a, and provides more context information to the model which facilitates the prediction of matching embeddings.

Given a fixed compute budget, *i.e.* size of the image in pixels, a trade-off is necessary between the total area covered by the image, and the Level of Detail (LoD) depicted in the image. A low pixel resolution yields a large area and overlap with the camera frustrum, but much less visible detail than the street-view photo. A high pixel resolution increases the LoD, albeit in a smaller overlapping region that excludes parts of the scene that are too far away from the camera location.

Our third and final strategy (*cf*. Fig. 4.3c) is based on the observation that the visible detail of objects in a street-view photo decreases with their distance to the camera. We conjecture that objects depicted with less detail in the street view also require less detail in the aerial view to successfully align their embeddings. Consequently, a high LoD of aerial imagery is only required in a small area around the cell where objects are also visible with high detail from camera locations in the cell. A lower LoD is sufficient for parts of the scene that are further away and cover a larger area.

We choose a set of n aerial images per cell with the same size in pixels that provide varying coverage and LoD of the scene around the cell. The model is given the n images and predicts a single embedding vector for the cell. We sample the first image with a side-length of d_0 meters to provide a high LoD in a small area around the cell. The side-length is doubled for each subsequent image resulting in lower LoD over a larger area:

$$d_i = 2^i d_0 \text{ for } i \in [0..n-1] \tag{4.3}$$

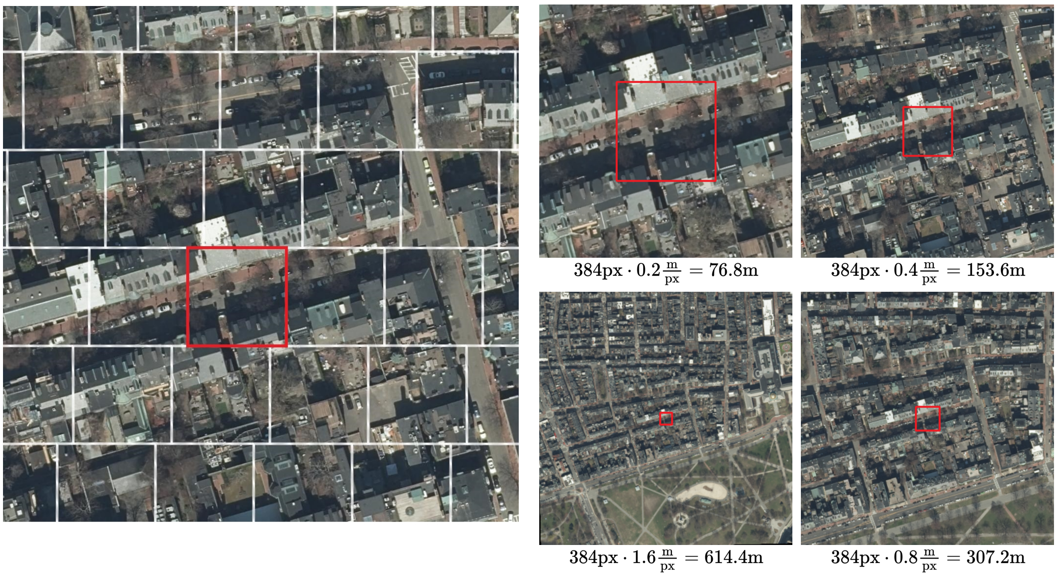
Fig. 4.4 shows an example of aerial images sampled for a search cell with a size of $384\text{px} \times 384\text{px}$ and resolutions of $0.2\frac{\text{m}}{\text{px}}$, $0.4\frac{\text{m}}{\text{px}}$, $0.8\frac{\text{m}}{\text{px}}$ and $1.6\frac{\text{m}}{\text{px}}$. A single aerial image that covers the same area with the highest pixel resolution of $0.2\frac{\text{m}}{\text{px}}$ would result in 16 times more pixels to process.

Comparison to VIGOR The choice of aerial images in VIGOR proposed by Zhu *et al*. [163] (*cf*. Sec. 2.2.2) is related to the above strategies as follows. Similar to the first strategy (*cf*. Fig. 4.3a), Zhu *et al*. implicitly adopt a coupled view of search cells and aerial images. If a street-view image is captured from any location on an aerial image, it is defined as a positive or semi-positive match. The size of the search cell is thus equal to the size of the corresponding aerial image.

To ensure that each potential camera location is covered by at least one aerial image with sufficient context information, Zhu *et al*. sample the aerial images with an overlap of half their side-length over the search region. However, due to the coupled view described above, each query location is thus also covered by multiple positive or semi-positive search cells. This necessitates the many-to-many formulation in VIGOR. While some matching query and reference views consequently have only a small overlap, the loss function used by Zhu *et al*. nevertheless forces the corresponding embeddings to partially align, and therefore results in noisy supervision.

Interestingly, both VIGOR and our second strategy (*cf*. Fig. 4.3b) sample the same overlapping aerial images over the RoI. The difference lies in the definition of the corresponding search cells, which are also *overlapping* in the many-to-many formulation, but smaller and *non-overlapping* in our many-to-one formulation.

Our proposed decoupling of aerial images and search cells for the first time facilitates the usage of aerial imagery at multiple LoD, and leads to our proposed third and final strategy described above.


384px · 0.2 m/px = 76.8m
384px · 0.4 m/px = 153.6m
384px · 1.6 m/px = 614.4m
384px · 0.8 m/px = 307.2m

(a) A cell in the search region layout.

(b) Aerial images that are passed to the model for the cell with 384×384 pixels and varying pixel resolutions in $\frac{\text{m}}{\text{px}}$.

Fig. 4.4: Our proposed choice of aerial images that are passed to the model to predict an embedding representation for a single cell. The images provide a LoD for the scene that decreases with the distance to the cell. This aligns with the visible detail of objects in the street-view photo that decreases with their distance to the camera location.

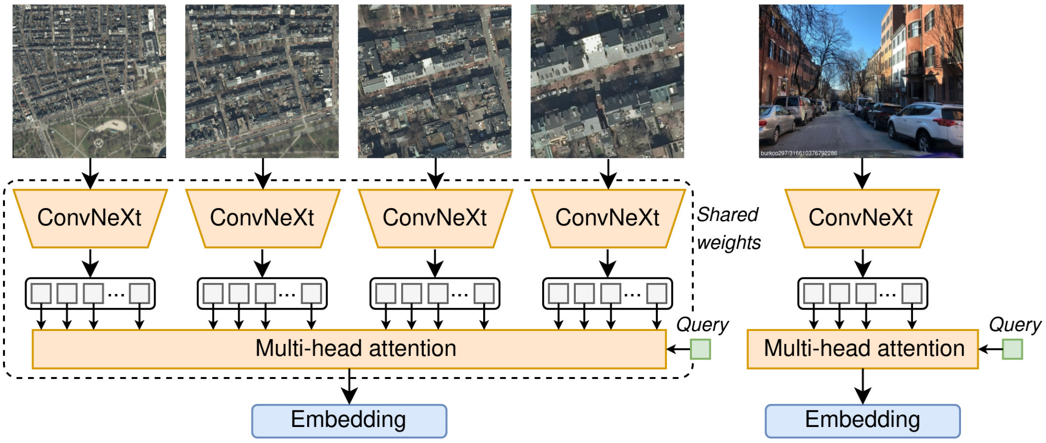
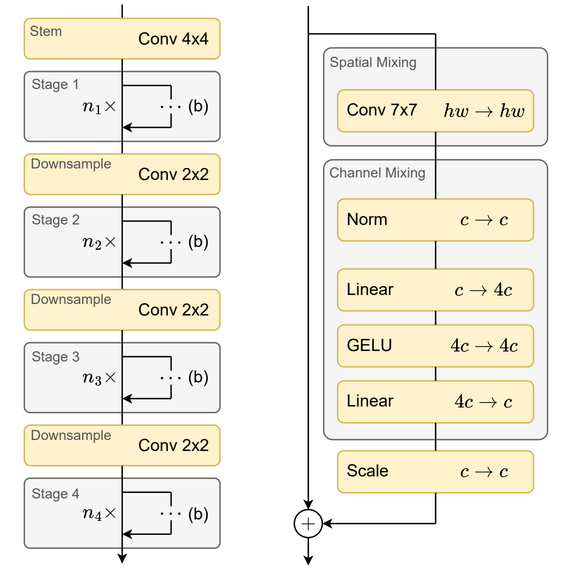


Fig. 4.5: Overview of the model architecture. We use a ConvNeXt backbone [80] to extract high-level features from the input images, and a Multi-head Attention (MHA) block [124, 69] with a single learnable query to aggregate features per image and across images into a single embedding representation.

4.3.3 Model Architecture

In this section, we provide a description of the model architecture that we employ to represent the embedding function for CVGL. We use one model to map a street-view image onto its embedding, and another model to map aerial images at multiple LoD onto the embedding of the respective cell.

To construct our model, we use general-purpose building blocks that have been demonstrated to be effective in a variety of other domains. We employ a ConvNeXt backbone [80], *i.e.* a modernized CNN, to extract high-level features from the input images. The output of the backbone is given as a set of features that tend to capture information only about a local neighborhood in the input image. To aggregate the features from a single image in the street-view domain and from multiple images in the aerial domain, we employ a Multi-head Attention (MHA) block [124, 69]. Fig. 4.5 provides an overview of the architecture. The details are described below.



(a) The network consists of repeated application of the main block (b) and downsampling layers.

(b) Main block that processes a feature map with height h, width w and c channels.

Fig. 4.6: Overview of the ConvNeXt architecture introduced by Liu *et al.* [80].

Encoder The ConvNeXt architecture is a general-purpose vision encoder that has shown strong performance in varying domains such as image classification, object detection and semantic segmentation [80]. The authors consider the earlier ResNet architecture [52] and make several modifications that are motivated by previous works [149, 32] and significantly improve performance while keeping compute approximately equal. Fig. 4.6 provides an overview. The network comprises the following processing steps.

The **stem** applies a convolution with 4×4 kernels and a stride of 4 to downsize the input image and increase the number of channels to $c_1 = 128$ in the base configuration. Stem layers are typically used in vision architectures to reduce the spatial dimensions of the input image and thereby the memory and computational requirements of the model.

The **block** is the main processing unit in ConvNeXt. It is separated into a *spatial mixing* part that communicates information only along the spatial dimensions of the feature map, *i.e.* for each channel independently, and a *channel mixing* part that communicates information only along the channel dimension, *i.e.* for each pixel independently. Spatial mixing is done using a convolution with one kernel per channel and a large kernel size. This puts ConvNeXt in the category of CNNs. Channel mixing constitutes the representational power of the block and contains most of the learnable parameters. It utilizes layer normalization [8] and a small two-layer MLP with the GELU activation function [54].

The block employs a residual connection [52] such that internal layers do not directly predict the output feature map, but rather a delta that is added onto the input features to yield the output feature map. The last layer in the block consists of a c-dimensional parameter that rescales (*i.e.* multiplies with) the delta predicted by the block. It is initialized with values close to zero, such that each block initially applies only small changes to the features and approximates an identity mapping. The scale per block increases during training as the block learns an operation that is useful towards minimizing the loss function.

Each **stage** represents the successive application of several main blocks that retain the dimensionality of the feature map. ConvNeXt is made up of four stages that employ $n_1 = 3$, $n_2 = 3$, $n_3 = 27$ and $n_4 = 3$ blocks in the base configuration. In between the stages, the spatial dimensions of the feature map are downsampled by a factor of 2, and the number of channels is increased by a factor of 2 using a convolution with 2×2 kernels. Each subsequent stage thus represents more abstract information at a lower spatial resolution.

ConvNeXt is introduced with several variants from *tiny* to *extra-large* based on the choice of c_1 and n_i. Larger variants typically outperform smaller variants, but have larger memory and computational requirements. The network was first employed in the context of retrieval-based CVGL by Deuser *et al.* [30] who report state-of-the-art results on benchmarks such as VIGOR. They train a single model that is applied to both the street-view and aerial domain and mean-pool the resulting feature map to yield the final embedding vector.

Aggregation layer Several methods have been introduced in the context of VGL to aggregate the feature map from a CNN into a single embedding representation for the image, including NetVLAD [7], SAFA [108] and SMD [165] (*cf*. Sec. 2.5). The model used by Deuser *et al.* employs a simple mean-pooling layer to aggregate local features and achieves state-of-the-art results; however, their work includes other advances such as the loss function and vision encoder that prohibit inspecting the merit of mean-pooling as aggregation layer ceteris paribus.

Our proposed problem formulation also requires aggregating features not only per image, but across multiple images in the aerial domain. We therefore develop a novel aggregation layer that is able to address this problem and outperforms both mean-pooling and more complex existing layers both in a single LoD and multiple LoD setting. Our layer uses MHA, a general-purpose building block with proven record in other domains such as image classification [32] and Natural Language Processing (NLP) [124]. The following paragraphs describe the concept of attention in neural nets, and how we use it as aggregation layer for our embedding function.

MHA was popularized by the Transformer architecture [124] and is used to allow for global communication between a set of tokens (*i.e.* feature vectors). In NLP, the tokens represent words in a text[1], while in our context they are the set of local features predicted by a CNN. Given n_q query tokens and n_v key and value tokens, MHA allows each query to aggregate information from the set of value tokens based on its similarity to the respective key tokens. The operation is described as *self-attention* if queries, keys and values originate from the same set of input tokens with $n_q = n_v$. It is described as *cross-attention* if queries come from a different origin as keys and values.

Let $Q \in \mathbb{R}^{n_q \times c}$, $K \in \mathbb{R}^{n_v \times c}$ and $V \in \mathbb{R}^{n_v \times c}$ be the features of queries, keys and values. The attention operation is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{c}})V \tag{4.4}$$

The product $QK^T \in \mathbb{R}^{n_q \times n_v}$ represents the attention score assigned by each query to each value and is determined via the dot-product similarity between queries and keys. The scores are normalized using a softmax operation such that each score is in the range $[0, 1]$, and the scores per query sum up to 1. For each query, a weighted average is computed over all values using a matrix multiplication with the normalized attention scores, and returned as the output of the attention operation. The number of output tokens is thus equal to the number of query tokens.

The attention operation computes a single weighted average over the value tokens. In *multi-head* attention with h heads, the feature matrices Q, K and V are instead split along the channel dimension into h equally sized sub-matrices, an attention operation is computed for each head, and the outputs are concatenated along the channel dimension to yield the final output features. This allows MHA to compute multiple weighted averages over the input features.

Our aggregation layer uses a MHA block to pool features both per image and across images as follows. We first apply layer normalization [8] to the output features of ConvNeXt, resulting in the set of tokens $X \in \mathbb{R}^{n_v \times c}$. In the street-view domain, the tokens originate from the single input image, while in the aerial domain they originate from multiple images and are concatenated along the token dimension of X. We define a single learnable query token $q \in \mathbb{R}^c$ that attends to the tokens X in

[1] More generally, each token represents a short sequence of characters that is determined for example using the Byte Pair Encoding (BPE) [41] and does not necessarily align with word boundaries.

a MHA block. We use $K = X$ and compute V via a learnable linear projection of X. The output of the MHA operation is L2-normalized and used as the embedding vector $f \in \mathbb{R}^c$ for the respective street-view image or search cell.

Summary Fig. 4.5 provides an overview of the architecture described above. The ConvNeXt backbone is used to extract high-level features from the input images, and the MHA block aggregates features both per image and across images into a single embedding representation. We use shared weights for the models applied to the different LoD in the aerial domain which reduces the number of learnable parameters and computational complexity. We do not use shared weights between the street-view and aerial domains as in Sample4Geo [30] due to their significant appearance changes.

4.3.4 Loss Function

We train the model on batches of matching pairs of street-view images and search cells analogous to CLIP [101] and Sample4Geo [30]. Instead of a cross-entropy loss, we employ the Decoupled Contrastive Learning (DCL) loss [154] which is introduced by Yeh *et al.* to address the usage of smaller batch sizes.

For each batch with b matching pairs, the images are processed by the model resulting in embeddings $F_s \in \mathbb{R}^{b \times c}$ for the street-view images and $F_a \in \mathbb{R}^{b \times c}$ for the search cells. We compute the dot-product similarity between all pairs of street-view images and search cells in the batch (*cf*. Fig. 4.7):

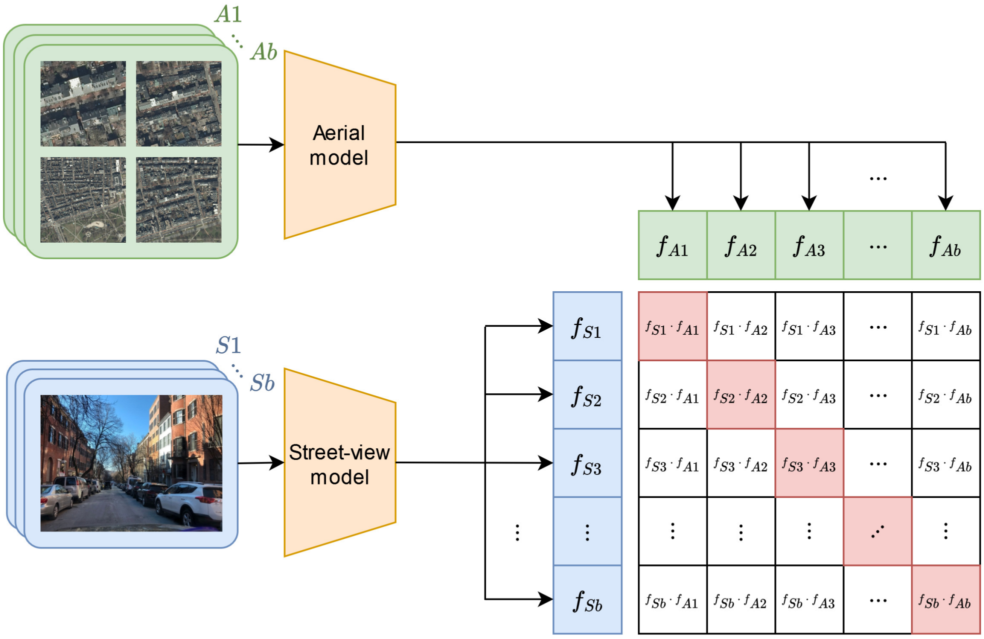
$$S = F_s F_a^T \in \mathbb{R}^{b \times b} \tag{4.5}$$

Each row and column in S represents a classification problem over b classes with one positive score and $b - 1$ negative scores. We apply the DCL loss with label smoothing to each row and column independently and compute total loss as the mean of all individual losses. The DCL loss without label smoothing is defined as

$$\mathcal{L}(s^+, s_1^-, ..., s_{b-1}^-) = -\ln \frac{e^{\frac{1}{\tau}s^+}}{\sum_i e^{\frac{1}{\tau}s_i^-}} \tag{4.6}$$

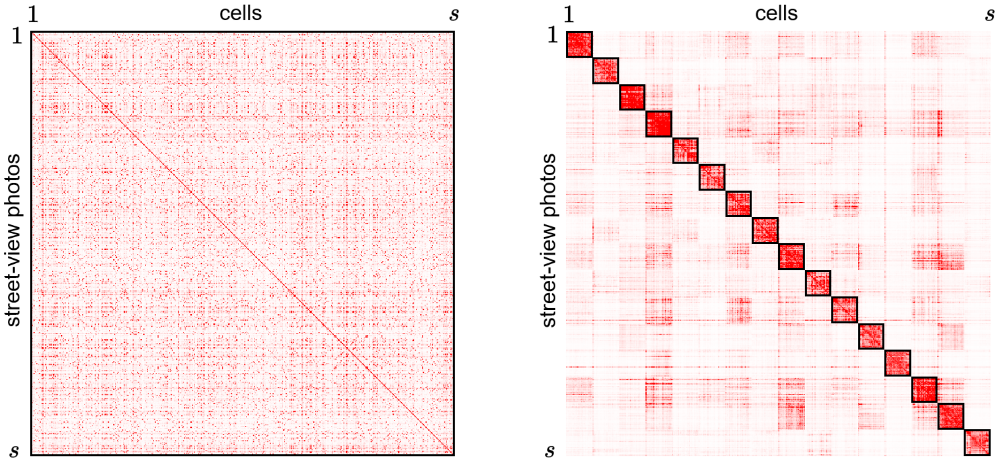
where s^+ is the similarity of the positive pair, and s_i^- the similarity of a negative pair. The DCL loss is similar to the InfoNCE loss [94] (*cf*. Eq. (2.4)), but removes the term corresponding to the positive pair in the denominator. We propose adding label smoothing to the DCL loss, resulting in the following expression:

$$\begin{gathered}\mathcal{L}(d_1, ..., d_b) = \sum_i -p_i \ln \frac{e^{\frac{1}{\tau}d_i}}{\sum_{j \neq i} e^{\frac{1}{\tau}d_j}} \\ \text{with } p_i = \begin{cases} 1-\epsilon & \text{if } d_i \text{ describes a positive pair} \\ \frac{\epsilon}{b-1} & \text{if } d_i \text{ describes a negative pair} \end{cases}\end{gathered} \tag{4.7}$$



	f_{A1}	f_{A2}	f_{A3}	⋯	f_{Ab}
f_{S1}	$f_{S1}\cdot f_{A1}$	$f_{S1}\cdot f_{A2}$	$f_{S1}\cdot f_{A3}$	⋯	$f_{S1}\cdot f_{Ab}$
f_{S2}	$f_{S2}\cdot f_{A1}$	$f_{S2}\cdot f_{A2}$	$f_{S2}\cdot f_{A3}$	⋯	$f_{S2}\cdot f_{Ab}$
f_{S3}	$f_{S3}\cdot f_{A1}$	$f_{S3}\cdot f_{A2}$	$f_{S3}\cdot f_{A3}$	⋯	$f_{S3}\cdot f_{Ab}$
⋮	⋮	⋮	⋮	⋰	⋮
f_{Sb}	$f_{Sb}\cdot f_{A1}$	$f_{Sb}\cdot f_{A2}$	$f_{Sb}\cdot f_{A3}$	⋯	$f_{Sb}\cdot f_{Ab}$

Fig. 4.7: Overview a training batch of size b with street-view images $S1$ to Sb and search region cells $A1$ to Ab represented by aerial images. The street-view and aerial models are used to encode input samples to the corresponding embedding vectors f. We compute the pairwise similarity matrix following CLIP [101]. Values on the diagonal correspond to positive matches, and all other values to negative matches.



(a) A large batch size s would ensure that all hard pairs of samples in the mining pool (red pixels) are in the same batch and supervised by the loss function, but is not computationally feasible.

(b) We instead partition the mining pool into clusters with size $b < s$. Each cluster is used as a batch in the training loop. Most hard pairs are contained in the same cluster and supervised by the loss function.

Fig. 4.8: Overview of our proposed HEM strategy. The images depict the $s \times s$ similarity matrix between s queries and s references in a mining pool during the late stage of a training. The color of pixels indicates the similarity of the respective pair between low (white) and high (red). Pixels on the diagonal represent positive pairs, and all other pixels represent negative pairs. Each black box marks a set of pairs that is contained in the same batch (*cf*. Fig. 4.7) and therefore supervised by the loss function.

4.3.5 Hard Example Mining

We find that the model tends to converge to a state early during training where it classifies most samples in most batches correctly and therefore receives little more supervision. This observation has also been made in previous works [55, 162, 30] (*cf*. Sec. 2.7). The difficulty of finding the correct reference cell for a given query increases with the number of cells that are under consideration. The size of a real-world reference database is typically much larger than the batch size b.

The problem is alleviated by choosing a larger batch size which ensures that more hard negatives are included per batch. For instance, Radford *et al*. [101] train a language-vision model with a batch size of 32k on 592 V100 GPUs. However, even at this scale the batch size is bounded by the available hardware. Several works follow a different approach by choosing a batch composition that includes more hard samples in batches with a smaller size b. This is called Hard Example Mining (HEM).

Existing HEM approaches consider a mining pool from which hard examples are drawn. The mining pool comprises either the entire training dataset [7, 162, 163, 160, 30] or a subset thereof [139]. The pool is searched for samples that are close in the embedding space and are therefore hard for the model to discriminate. The embeddings for all samples in the pool are kept in memory and updated periodically during training as the model's performance improves.

We consider a semi-infinite data setup where the dataset is large enough for each street-view photo to be seen only once, and the training thus encompasses only a single epoch. HEM strategies that keep in memory the embeddings of the entire dataset are therefore not applicable. We propose a novel HEM strategy as follows.

To perform HEM at some point during training, we consider the next s randomly drawn examples as mining pool. We predict the embeddings for all examples in the pool using forward passes with the current model state. A single batch with size s would ensure that the loss is computed for all hard pairs of samples in the pool, but require unfeasible memory and computational resources with large s (*cf*. Fig. 4.7a). Instead, we partition the pool into $\frac{s}{b}$ batches with batch size b such that each batch ideally contains many hard pairs of samples, and easy pairs that have only small contribution to the loss are split across different batches (*cf*. Fig. 4.7b).

We address the partitioning problem using a simple, but computationally effective clustering approach. To construct a cluster, we first draw a random sample (*i.e.* positive query-reference pair) from the mining pool as the cluster's seed. We iteratively select additional samples whose reference embedding is closest to the centroid of the cluster's query embeddings. We update the centroid after each sample is added. The process is repeated until all samples in the pool are assigned to a cluster. After the HEM is done, the clusters are forwarded as batches to the original training loop.

Our proposed approach represents a novel *clustering-based* view on HEM that aims to maximize the hardness associated with each batch. In contrast, existing approaches interpret HEM as a *nearest neighbor* problem that maximizes the hardness associated with individual queries, and combines multiple queries to form a batch.

Our approach further allows for a novel interpretation of HEM as simply approximating the use of a larger batch size that spans the entire mining pool (*cf*. Fig. 4.8). The goal of the clustering method is to partition the pool into batches such that pairs of samples that are spread across multiple batches have only a small contribution to the total loss associated with the larger batch size.

The hardness of batches that are constructed with our clustering-based HEM depends on the performance of the model and the size of the mining pool:

1. Given a fixed size s, the recall per batch increases as the model's performance increases during training; batches that are drawn from the same distribution become easier for the model to solve. If s is chosen too small (*e.g.* $s = b$ corresponding to randomly sampled batches), the model is faced with a lack of supervision at some point during training.

2. Given a fixed model state, samples per batch are harder for the model to discriminate if they are drawn from a larger mining pool. This is analogous to increasing the hardness of batches by choosing a larger batch size, but is not bounded by the available GPU resources.

We start the training without HEM, *i.e.* with $s = b$. Since the model's parameters are initialized randomly, most pairs are classified incorrectly and have a significant contribution in the loss. As the training progresses, the hardness of batches decreases due to the improving model performance. We increase s periodically to compensate for this effect and keep the per-batch recall low. We choose a training schedule for s that doubles the size after every mining pool and at least $\frac{5000}{b}$ samples.

Increasing the size of the mining pool during training has several advantages over using a constant pool size:

Rate of change The model parameters change rapidly early during training and become more stable as the training progresses. A high rate of change of model parameters requires a smaller pool size to avoid creating batches that are based on outdated embeddings towards the end of each pool. A low rate of change allows for larger pool sizes since embeddings remain valid for more training iterations. Our increasing pool size reflects the decreasing rate of change of model parameters during training.

Curriculum learning A general-purpose training strategy, called *curriculum learning*, suggests that training data should not be shown to a model in random order, but instead start with easy examples and gradually increase in complexity (*i.e.* hardness) [11, 137]. Our HEM strategy represents a training curriculum that aligns with this idea. We start with randomly sampled batches that represent the easiest type of training example, and gradually increase the hardness by sampling batches from larger mining pools.

Computational complexity The cost of predicting embeddings for a mining pool with size s is in $\mathcal{O}(s)$. Since a new mining pool is considered only every s iterations, the additional cost per iteration is in $\mathcal{O}(\frac{s}{s}) = \mathcal{O}(1)$, *i.e.* independent of the choice of s. However, the complexity of the clustering method is in $\mathcal{O}(s^2)$, and the additional cost per iteration therefore is in $\mathcal{O}(\frac{s^2}{s}) = \mathcal{O}(s)$. The total time that is spent on HEM during training thus increases with the size of the mining pool s. Using a smaller s early during training reduces this cost.

In summary, our proposed HEM strategy shifts from optimizing the hardness of individual queries via a nearest neighbors method to optimizing the hardness of individual batches via a clustering-based method. This allows for a novel interpration of HEM as simply approximating the use of a larger batch size in contrastive training settings.

4.3.6 Data Augmentation

Each training batch contains b street-view images sampled randomly from our dataset, as well as b matching search region cells. For each street-view photo, we construct a *virtual* matching cell around the photo's geolocation, rather than finding the corresponding cell in our search region layout. The cell is chosen with a random orientation in $[0, 2\pi)$ and translational offset $t \in [-t_{\text{max}}, t_{\text{max}}]^2$ with $t_{\text{max}} = 0.5l - \Delta l$.

The random orientation serves as a method for data augmentation, for instance to generalize across the direction of shadows in the scene: If the aerial imagery in the train and test regions are captured during different times of the year in different geographical locations, the diverging direction of shadows could otherwise potentially result in a domain gap.

The random translation ensures that the camera's geolocation is at least Δl away from the boundary inside the cell which partially compensates for GNSS ground-truth errors in the dataset. We choose $\Delta l = 5.0\text{m}$. We further apply a simple color augmentation to all images during training.

4.4 Data

Existing datasets in the field of retrieval-based CVGL are subject to limitations on the applicability under real-world conditions and scalability to large geographical regions. These limitations include the non-transferability of the one-to-one setting to a densely covered RoI (*cf*. Sec. 2.2.1), the inconsistent scale of aerial images and search cells due to the overlooked utilization of a Mercator's projection (*cf*. Sec. 2.4), the inability for research works to investigate the search region layout and scale of aerial images due to providing a predefined set of orthophotos, and a reliance on known compass orientation (*cf*. Sec. 2.3.2) or panoramas that provide a $360°$ FoV.

We therefore collect a new dataset to train and evaluate our proposed method. The data is gathered from publicly available sources in Germany and the US in a much larger and more diverse scope than existing datasets. It is provided in a format that allows defining and investigating varying search region layouts, and provides an interface to sample aerial imagery at any required location, orientation, scale and size. The following sections describe the details of the data and collection process.

4.4.1 Aerial Data

We gather aerial imagery from several states in Germany and the US that provide statewide, recent orthophotos at high resolution. The states are: Berlin and Brandenburg [42], Massachusetts [86], North Carolina [92], North-Rhine Westphalia [96], Saxony [43] and Washington D.C. [95] (*cf*. Tab. 4.1 and Fig. 4.9). The data are published by the states following open data policies and are therefore subject to permissible licenses. Overall, our dataset contains aerial imagery over an area of $\sim$246000 km^2 with a total size on disk of ~ 6.5TB.

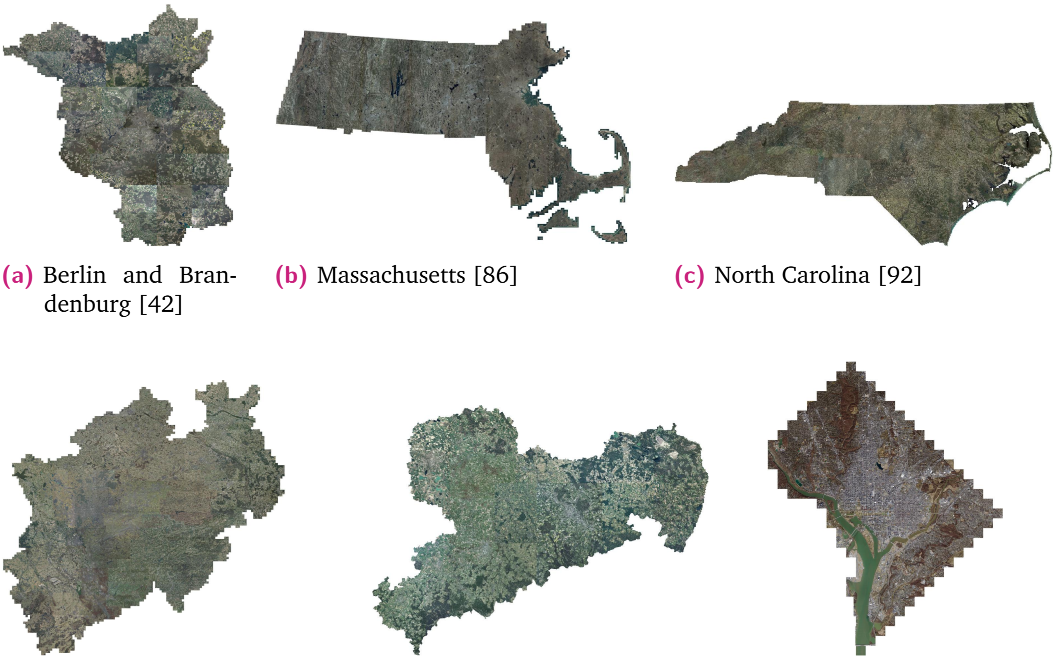
Aerial imagery is provided by all seven states as tiled orthophotos that represent a flattened view of the globe following a chosen map projection. Each individual tile covers a large area, *e.g*. 10000×10000 pixels in Massachusetts. We download all tiles from the provider and store them for further processing.

When an aerial image with a specific location, orientation, scale and size is requested by the training loop, we load only those tiles from disk that have an overlap with the requested region. The tiles are then merged, transformed according to the given parameters and rescaled to compensate for the underlying map projection. To speed up this step, we implement two optimizations:

1. We partition the original, large tiles into smaller tiles with a resolution of 250×250 pixels and store the smaller tiles on disk. Since all experiments in our work use aerial images with a size between 256×256 and 512×512 pixels, this avoids having to load large tiles from disk of which only a small portion is actually required.

Tab. 4.1: Overview of the aerial imagery included in our dataset. The pixel resolution varies slightly across the regions due to changes in scale incurred by the map projection.

State(s)	Area	Resolution ($\frac{\text{m}}{\text{px}}$)	Memory (TB)	
	(10^3 km^2)		raw	jpeg
Berlin, Brandenburg [42]	32.2	~0.20	2.4	0.3
Massachusetts [86]	22.8	~0.15	3.0	0.7
North Carolina [92]	135.2	~0.15	17.4	4.0
North Rhine-Westphalia [96]	35.8	~0.10	10.8	1.3
Saxony [43]	19.9	~0.20	1.5	0.2
Washington D.C. [95]	0.2	~0.08	0.1	<0.1
Total (Σ)	246.1	–	35.2	6.5


(a) Berlin and Brandenburg [42]

(b) Massachusetts [86]

(c) North Carolina [92]

(d) North Rhine-Westphalia [96]

(e) Saxony [43]

(f) Washington D.C. [95]

Fig. 4.9: Overview of the aerial imagery included in out dataset. Images for Berlin and Brandenburg are provided by a joint service. Not to scale.

2. We precompute downsampled versions of the imagery and store them on disk at different zoom levels. The zoom level $z \in \mathbb{N}$ corresponds to a downsampling factor of $\frac{1}{2^z}$. When requesting images at a low pixel resolution (*e.g.* the aerial image with the lowest LoD for a given cell), this avoids having to load many high resolution tiles from disk that are only used in downsampled form.

Our online sampler represents an interface to the aerial imagery that abstracts from the underlying tiling scheme, map projection and optimization mentioned above. The interface is defined simply as a function that maps from image parameters to the requested image in a given region:

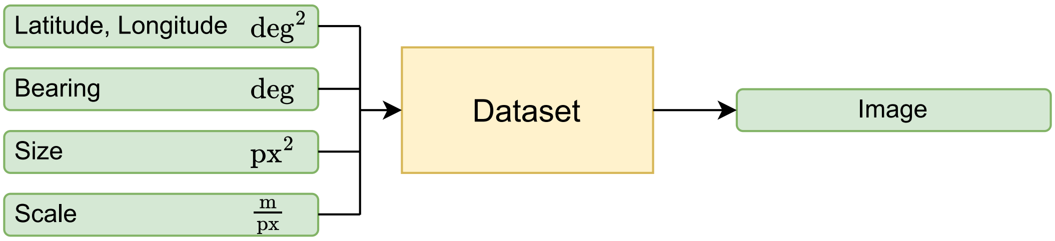

Latitude, Longitude deg^2
Bearing deg
Size px^2
Scale $\frac{\text{m}}{\text{px}}$
Dataset
Image

Fig. 4.10: Interface to retrieve aerial imagery in our dataset.

In contrast, existing datasets provide aerial imagery as a predefined set of images and associated metadata:

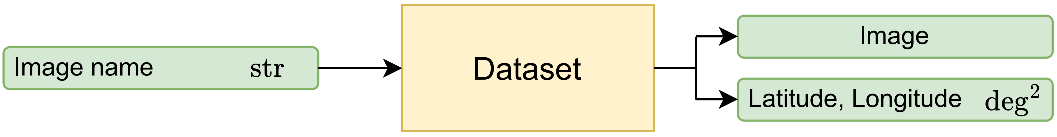


Fig. 4.11: Interface to retrieve aerial imagery in existing datasets.

The main advantages of our interface are as follows:

- The location, size and scale of images in existing datasets is predefined according to a specific search region layout. There is no interface to retrieve images with other parameters. In contrast, our dataset allows sampling images with any requested parameters and for the first time facilitates research into other search region layouts and the usage of aerial images at multiple LoD.
- All images in existing datasets are north-aligned. Other bearings require manual rotation of the image about its center. Any bearing that is not a multiple of 90° (*e.g.* for the purpose of data augmentation) results in parts of the image being cropped. Images in our dataset are directly sampled from the underlying tiled data and do not contain cropped regions.
- The scale of aerial images, *i.e.* meters per pixel, in existing datasets is both (1) inconsistent and (2) implicit, *i.e.* not reported as metadata. This has been

overlooked in existing works, and has resulted among others in incorrect ground-truth annotations [163, 71] and an unexplained domain gap between different geographical regions [35]. The explicit scale parameter in our interface provides a simple means to ensure consistent scale across regions, and compensates for the inconsistency of the underlying map projection. Requesting a varying scale of images for any reason is still possible, but requires an explicit acknowledgement by setting the scale parameter accordingly.

4.4.2 Street-view Data

To build our dataset, we require geo-referenced street-view images that are captured from the regions for which aerial imagery is available. We use street-view images to train and evaluate our method in this work; however, they are not required during real-world deployment in a RoI.

We collect street-view images from the crowd-sourcing platform Mapillary [85] which provides data for most regions across the globe in large quantity and diversity. While the density of images on the platform in large part is not sufficient to employ SVGL approaches, it nevertheless allows training and testing CVGL methods which rely only on densely available aerial imagery. We download all images from Mapillary that are located in regions where our dataset contains aerial imagery. The process is implemented as follows.

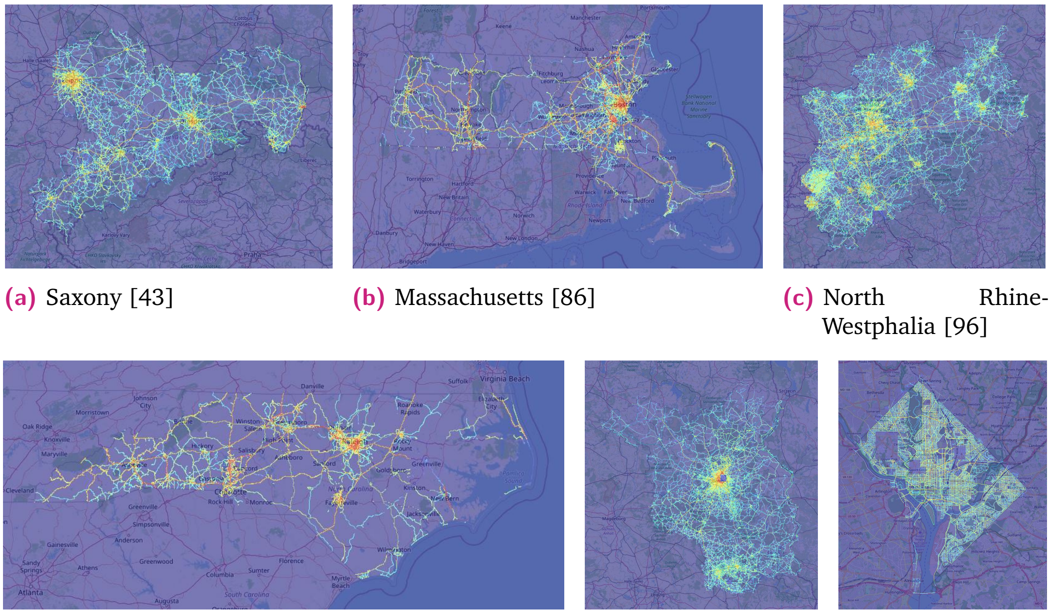
The Mapillary service provides an Application Programming Interface (API) that allows querying images in a region defined by a bounding box of latitude and longitude values. Each query returns at most 2000 images regardless of the size of the region. We implement a recursive strategy that starts with a large bounding box covering the entire RoI, and splits a box into four smaller boxes if the respective query returns the maximum of 2000 images. If less than 2000 images are returned for a bounding box, it is removed from the processing queue and the respective images are stored in the dataset. We download only photos with limited FoV and discard panoramas. All images are resized and padded to 640×480.

Fig. 4.12 and Tab. 4.2 provide an overview over the resulting geographical locations of street-view images downloaded from Mapillary. Overall, our dataset contains 72.7M street-view images spread across 1.5M cells of size 100m×100m. The number of cells allows approximating the portion of the aerial imagery that is usable during training: Street-view images that are located in close proximity to each other are paired with largely overlapping aerial images.

Fig. 4.13 shows randomly chosen street-view images from the regions that make up our dataset. Most images are captured on roads and have a forward perspective. We find that images in the US are focused almost exclusively on major routes, covering

Tab. 4.2: Overview of the street-view images included in our dataset. # Cells indicates the number of 100m×100m cells (*cf*. Sec. 4.3.1) that include at least one street-view image. The *coverage of aerial imagery* indicates the percentage of aerial imagery in the respective state(s) that is covered by cells with street-view images.

State(s)	# Images	# Cells	Coverage of aerial
	(10^6)	(10^3)	imagery (%)
Berlin, Brandenburg	17.8	350	10.9
Massachusetts	10.7	124	5.4
North Carolina	9.4	227	1.7
North Rhine-Westphalia	21.0	553	15.4
Saxony	6.1	214	10.8
Washington D.C.	7.8	15	68.3
Total (Σ)	72.7	1483	6.0



(a) Saxony [43]

(b) Massachusetts [86]

(c) North Rhine-Westphalia [96]

(d) North Carolina [92]

(e) Berlin and Brandenburg [42]

(f) Washington D.C. [95]

Fig. 4.12: Overview of the geographical locations of street-view images downloaded from Mapillary for our dataset. The color indicates the number of photos per pixel between 0 (blue) and the relative maximum in the image (red). Not to scale.

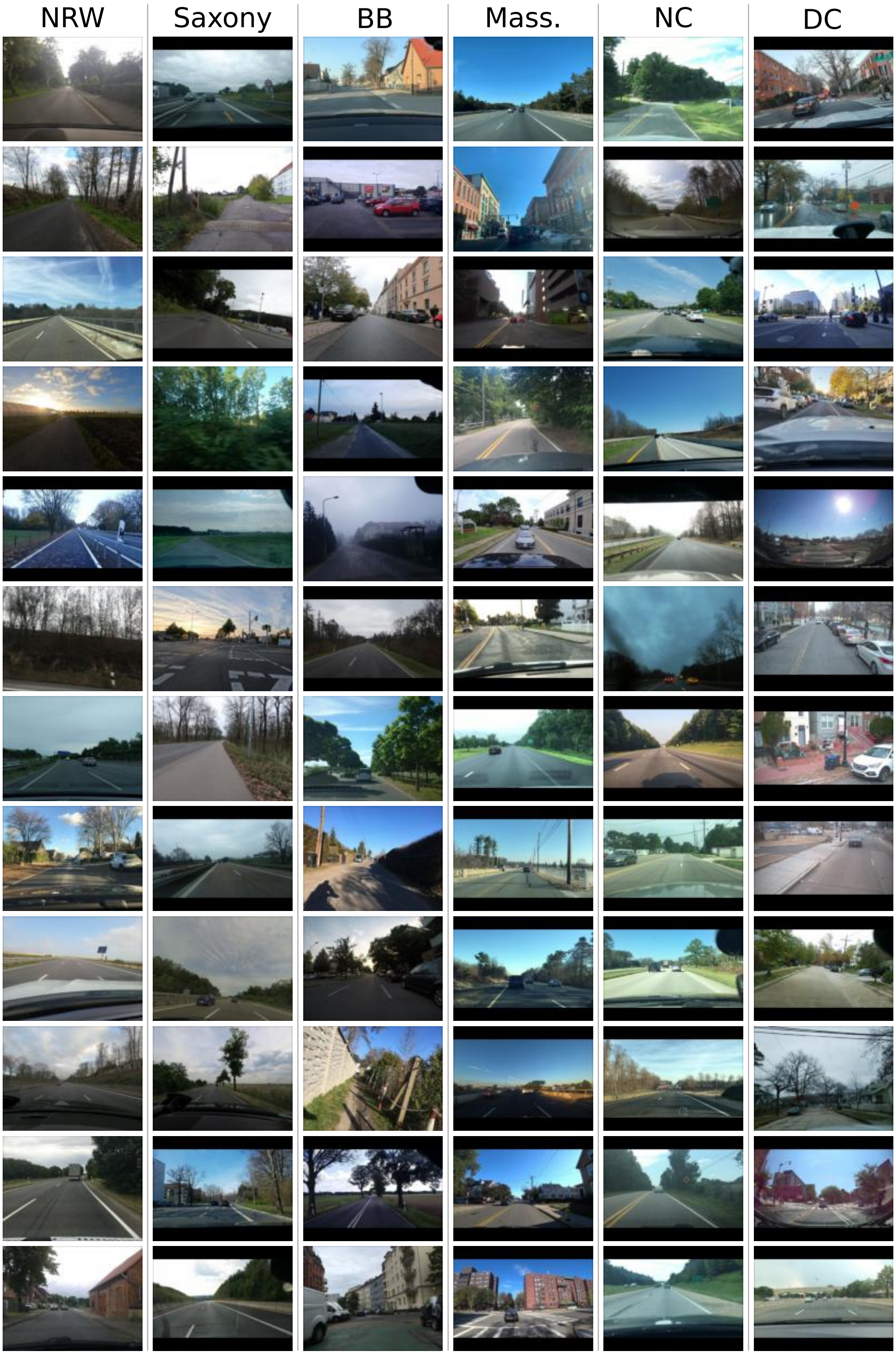


Fig. 4.13: Randomly chosen street-view images downloaded from Mapillary for the regions that make up our dataset.

1-5% of the respective aerial imagery (excluding Washington D.C. which is a smaller, mostly urban region). Photos in the German states have a larger diversity and cover 10-15% of the aerial imagery.

Photos on Mapillary are captured and uploaded by different individuals around the world under real-world conditions and often with consumer-grade devices. In contrast, most existing datasets in the field of CVGL represent *clean* data that are captured with dedicated devices (*e.g*. Google Street-view vehicles) and undergo filtering and postprocessing steps to improve the quality for instance of the geolocations and orientations. The challenge posed by our data however is also its main strength: Training a model on a diverse range of real-world photos improves the robustness w.r.t. real-world uses cases where images are not captured under ideal conditions either. On the other hand, training a model on clean data results in a significant domain gap when testing on real-world data with other, more severe types of image noise.

The street-view images downloaded from Mapillary offer a large diversity among others w.r.t. the following parameters:

- Weather conditions: *E.g*. rain, fog, snow, sun.
- Seasonal variations: *E.g*. change of vegetation.
- Daylight variations.
- Camera models: Intrinsic parameters and lens distortion.
- Camera orientation: Roll and pitch angle. (The yaw angle points mostly in the direction of travel).
- Platform: *E.g*. vehicles, bicycles, pedestrians.
- Image Noise: *E.g*. out-of-focus, motion blur.
- Occlusion: *E.g*. other vehicles, own dashboard or windshield wiper.

We find that the metadata associated with images on Mapillary, such as the camera orientation w.r.t. a geo-coordinate system, are not always accurate. We discard all metadata aside from the geolocation and perform retrieval based only the raw RGB image. This requires the model to generalize over parameters such as the camera model without additional input at test time or auxiliary ground-truth at training time. It greatly simplifies deployment of the model, especially in a post hoc setting where additional metadata are not always available. Furthermore, discarding parameters such as the compass orientation avoids any ambiguity on whether they may be passed to the model at test time (*cf*. Sec. 2.3.2).

4.4.3 Dataset Splits

We partition the data into training and test splits as follows.

Train We train the model on data from Berlin, Brandenburg, North Carolina, North Rhine-Westphalia, Saxony and Washington D.C. Since the training is done on matched pairs of street-view and aerial images, it requires aerial imagery only for those regions in the states for which street-view images are available on Mapillary. This corresponds to roughly 1.7% of the aerial imagery in North Carolina, and 10-15% in the German states (*cf*. Tab. 4.2). Unlike in SVGL, the training is independent of the density of street-view images: Even an isolated photo with no other photos nearby is usable.

Images on Mapillary are distributed non-uniformly across geographical regions. For instance, Fig. 4.12 shows that the density is typically much higher in urban regions. The city state of Washington D.C. alone contains 10.7% of the street-view images in our dataset, but corresponds only to 1.0% of the geographical cells. Choosing images randomly during training would result in oversampling of regions with a high density of street-view images, and potentially result in overfitting of the model w.r.t. to the corresponding aerial images. Instead, we randomly sample from the distribution of cells rather than images, and draw a single random photo per cell to be used in the training loop.

Test The model is tested on data from the remaining state, *i.e*. Massachusetts. This ensures that unlike in SVGL there is no overlap between the training and test regions, and the evaluation therefore focuses on the zero-shot performance of the model to new regions. We define the RoI to cover the entire state, and therefore utilize all of the aerial imagery available for Massachusetts. The RoI is partitioned into roughly 25M search cells according to the layout described in Sec. 4.3.1.

We use all 10.7M street-view images available on Mapillary as queries in the test split. We do not filter out potentially erroneous images or perform other preprocessing to clean the data. The test split thus follows a distribution of the data that is to be expected under real-world conditions with consumer-grade devices. While other datasets such as SF-XL [12] manually choose a small set of queries as test split, such a *clean* data distribution does not reflect the performance that is to be expected under real-world conditions.

Test - Ablation studies We define a smaller test split as a subset of Massachusetts to allow for evaluation with a reduced computational complexity in the ablations studies. The split covers an area around Boston of roughly $\sim 100\text{km}^2$ and contains 100k street-view images as queries, similar in size to VIGOR [163].

4.4.4 Comparison with Other Datasets

Tab. 4.3 provides a comparison with other datasets in the fields of CVGL and SVGL. We make the following observations:

Tab. 4.3: Overview of datasets in the field VGL. # Cells indicates the number of 100m×100m cells that contain at least one street-view photo, and approximates the geographical coverage of the street-view data. The size of the test region is stated as the coverage of aerial imagery in many-to-many and many-to-one CVGL, as ✗ in one-to-one CVGL, and as the number of cells times the cell size in SVGL.

	# Images	# Cells	Videos	Test region	Consistent scale
Cross-view					
Ours	72.7M	1.5M	✓	22922.3km^2	✓
CVUSA [144]	44k	43k	✗	✗	✗
CVACT [77]	138k	12k	✗	✗	✗
Vo & Hays [130]	450k	-	✗	✗	✗
VIGOR [163]	105k	12k	✗	114.9km^2	✗
SeqGeo [155]	119k	10k	✓	✗	✗
Gama [132]	53.9M	65k	✓	✗	✗
Street-view to street-view					
St Lucia [88]	33k	<1k	✓	<4.8km^2	-
NCLT [21]	3.8M	<1k	✓	0.4km^2	-
Pittsburgh 250k [7]	274k	<1k	✗	0.9km^2	-
TokyoTM/247 [120]	288k	<1k	✗	4.2km^2	-
SF Landmarks [24]	1.1M	<1k	✗	1.3km^2	-
SF-XL [12]	41.2M	11k	✗	97.0km^2	-

- Our dataset covers a test region that is more than 100× larger than in any existing dataset. Most CVGL datasets represent a one-to-one matching problem and do not consider a densely covered RoI. VIGOR [163] and SF-XL [12] provide the largest existing test splits and are focused on cities in the US.
- The geographical coverage (*i.e.* number of cells) in our dataset is 25× larger than in any existing dataset. While Gama [132] and SF-XL [12] contain a similar number of street-view photos that are usable during training, they are spread over smaller regions and therefore cover much less aerial imagery.
- Our dataset is the first to compensate for the underlying map projection that is used by orthophoto providers and allows sampling aerial images at a consistent scale across geographical regions.

4.4.5 Reproducibility

The data are hosted by the original providers, *i.e.* Mapillary and the orthophoto repositories of the states in Germany and the US. We implement and publish download scripts that allow accessing this data. The scripts retrieve the aerial imagery, as well as the street-view images on Mapillary that are available in a specific geographic region.

We take the following measures to best facilitate the reproducibility of experimental results presented in this work (*cf*. Sec. 4.5). We choose a test region (*i.e.* Massachusetts) that keeps aerial imagery from the year 2021 available even if more recent imagery is added to the platform. Some orthophoto providers, such as in

Saxony [43] and Berlin-Brandenburg [42], allow batch downloading only the latest imagery which is updated every few years.

We provide the list of all street-view image identifiers for the Mapillary platform that were used in this work which allows downloading the photos even when more images are added to the service in the respective regions. However, the crowd-sourcing users on Mapillary retain rights to the uploaded material and are able to remove images from the platform. We thus do not guarantee that all images used in this work will be available in the future.

4.5 Evaluation

In this section, we report experimental results of our method on the dataset described in Sec. 4.4. We first provide details on our implementation of the proposed method and baselines, and define the metric that we use to measure the localization accuracy. We then present quantitative and qualitative results, and further investigate factors that influence the model's performance in the ablation studies.

4.5.1 Implementation

The following sections provide details on the implementation of the method described in this chapter.

Main We train our main model with the *Base* variant of ConvNeXt [80] which is pretrained on ImageNet [29]. We use $h = 64$ heads in the MHA layer to pool the tokens encoded by the CNN into an embedding representation with 1024 channels.

The model is trained for 200k iterations with a batch size of 30. The model therefore sees a total of 6M distinct street-view photos during training. We train with the Adam optimizer [63] and a learning rate of $1.0 \cdot 10^{-4}$, 1k iterations of linear warmup and a cosine decay schedule towards a minimum learning rate of $1.0 \cdot 10^{-5}$. The loss function is parametrized with a temperature of $\tau = \frac{1}{36} \approx 0.03$ and label smoothing [44] of $\epsilon = 0.1$. We clip gradients to a maximum global norm of 1.0 [97] and apply decoupled weight decay with a magnitude of $1.0 \cdot 10^{-2}$ [81].

We choose search region cells with size 30m×30m and pass four aerial images with resolution 384×384 to the model per cell with $d_0 = 384\text{px} \cdot 0.2\frac{\text{m}}{\text{px}} = 76.8\text{m}$ and $d_3 = 2^3 \cdot d_0 = 614.4\text{m}$. The HEM pool is capped at a maximum size of $s_{max} = 2^{14}$.

Ablations studies We choose an experimental setup for the ablation studies with reduced computational cost to allow running more evaluations. We use the *Small* variant of ConvNeXt as encoder in the model architecture. Aerial images are used with a smaller resolution of 256×256, and street-view images with a resolution of

320×240. The HEM pool is capped to $s_{max} = 2^{12}$ samples. All other parameters are kept as in the main evaluation if not stated otherwise.

Nearest neighbors To evaluate the model in a geographical region, the embeddings for all search cells are computed and stored as the reference database. To allow for an efficient retrieval from this database, we employ an approximate nearest neighbors algorithm implemented in the FAISS library [33]. FAISS provides varying algorithms aimed at different sizes of reference databases and retrieval performance. We use a method recommended for databases in the order of 1M reference samples[2], *i.e.* Hierarchical Navigable Small Worlds [84].

Baselines We reimplement several existing methods as baselines for the evaluation, *i.e.* SAFA [108], TransGeo [160] and Sample4Geo [30]. We make the following adaptations to align them with our problem setup and allow for a fair comparison:

- Sample4Geo uses the same vision encoder, *i.e.* the *Base* variant of ConvNeXt, as in our work. For fair comparison, we replace the VGG backbone [23] in SAFA with the more modern ConvNeXt. TransGeo uses a vision transformer as backbone; we use its *Base* variant which has a similar number of parameters as the analogous variant of ConvNeXt.
- We train and evaluate SAFA, TransGeo and Sample4Geo using the search region layout and choice of aerial imagery as in VIGOR [163]. We use a Mercator-based partitioning of the search region into cells (*cf.* Fig. 4.1b) that changes the cell size w.r.t. to the cell's latitude. The size of cells is 30m×30m at the equator. We pass a single aerial image to the model that has twice the side-length of the respective cell.
- We use the same optimizer and learning rate schedule as in our work. We reduce the learning rate to $3.0 \cdot 10^{-5}$ in TransGeo which uses a vision transformer as backbone. Since Sample4Geo relies on a larger batch size, we increase it from 30 to 90, train for a third of the iterations and use a larger learning rate of $3.0 \cdot 10^{-4}$ following the linear scaling rule [46]. For fair comparison, we do not employ sharpness-aware minimization [67] of TransGeo [160] in the baselines or in our method. It improves the optimization, but doubles the computational cost of a training run.
- TransGeo and Sample4Geo use HEM strategies that are designed for a dataset with a limited, small size such as VIGOR. Due to the semi-infinite data setup in our case, we use our HEM strategy instead, but adapt it as follows. For TransGeo we use a cluster size of 2 and combine 15 independently sampled clusters in a single batch. SAFA is trained without HEM.
- We train TransGeo without their proposed method for non-uniform cropping. The cropping requires the activation maps of all samples in the dataset to be

[2] `https://github.com/facebookresearch/faiss/wiki/Indexing-1M-vectors`

stored on disk, which incurs a much higher cost in our semi-infinite data setup than when training on a smaller dataset such as VIGOR. Furthermore, the relative improvement of employing the cropping technique is small, *e.g.* 59.8% to 61.5% on the VIGOR same-area split.

4.5.2 Metric

Methods for CVGL are typically evaluated using the *recall* metric. The recall R@k is defined as the percentage of queries where the matching cell is one of the top k retrieved cells from the reference database. R@1 thus defines the proportion of predictions that are correct.

In existing works, the recall is used to evaluate methods both in the one-to-one and many-to-many matching setup. However, we identify several disadvantages of this metric in the many-to-many (*i.e.* VIGOR [163]) or many-to-one (*i.e.* ours) settings where each embedding in the reference database covers a non-zero area of potential street-view camera locations:

1. A small error in the implicitly estimated location of a street-view photo results in a misassignment to a neighboring cell if the photo is located close to a cell boundary, but not if it is located in the center of a cell and away from the boundaries. The R@k metric penalizes only the former case, although the underlying error in the estimated location might be identical. The error further changes based on the chosen discretization of the search region into cells.

2. Choosing a larger cell size results in fewer total cells in the reference database and therefore an improved R@k metric. However, the metric should ideally report the localization accuracy independent of how the search region is partitioned into cells.

3. R@k is not robust to errors in the ground-truth locations of the dataset which result in incorrect assignment of street-view photos to search region cells.

To address these disadvantages, we propose a novel metric for CVGL, *i.e.* R@k<r, that measures the percentage of queries where any cell in a radius r around the ground-truth location is one of the top k retrieved cells from the database. R@k<r is more robust to the cell discretization and noise in the ground-truth locations than R@k, and reports roughly the same value for all cell sizes smaller than r.

Unless stated otherwise, we report the recall with $r = 50\text{m}$. This allows for example for the subsequent application of cross-view pose estimation methods (*cf*. Chapter 5). Similar metrics are used in the context of SVGL [12, 122] where a retrieved reference photo is counted as a match if it is within a radius r to the query photo's location.

Tab. 4.4: Recall on the Massachusetts test split in percent. The baselines are reimplemented as described in Sec. 4.5.1.

	R@1<50m	R@10<50m	R@100<50m
SAFA [108]	0.5	2.7	10.7
TransGeo [160]	2.8	10.8	28.3
Sample4Geo [30]	12.8	32.2	55.8
Ours	**60.6**	**75.5**	**84.1**

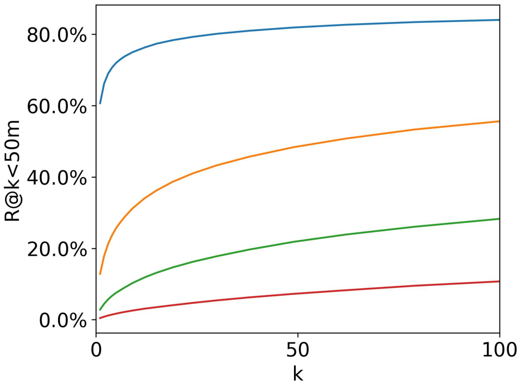


Fig. 4.14: R@k<50m on the Massachusetts test split for different values of k.

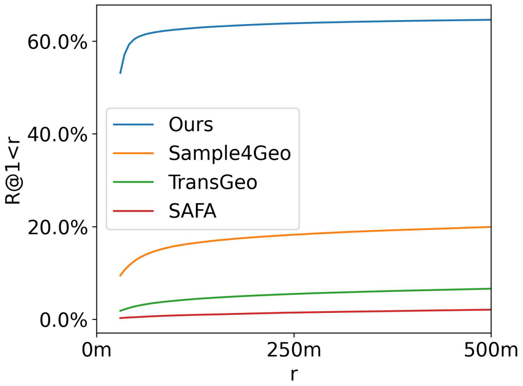


Fig. 4.15: R@1<r on the Massachusetts test split for different values of r.

4.5.3 Results

We report the main results of our work and baselines on the Massachusetts test split in Tab. 4.4. Our method correctly localizes 60.6% of all (non-panoramic) photos in Massachusetts to within 50m of their ground-truth location, without access to street-view data from this region for training or testing. The best existing method Sample4Geo [30] achieves a much lower recall of 12.8%. The reference database has a size of ~ 104GB and allows querying an embedding in less than 10ms using FAISS [33].

Fig. 4.14 shows the results for different values of k. Our method achieves a recall of 84.1% for $k = 100$. That is, for 84.1% of queries the camera location is confined to 100 potential cells (*i.e.* 0.0004% of the possible reference cells) or a 50m radius around them.

Fig. 4.15 shows the top-1 recall for different values of r. The recall increases when considering more cells around the camera locations as correct matches. However, the increase levels off at around 100m which suggests that for most images the error in the ground-truth location is below 100m.

Fig. 4.17 shows example images and the predicted top-1 locations and probability distribution over cells. Our model successfully localizes images even under varying illumination, weather and seasonal conditions, occlusion, image noise, and in

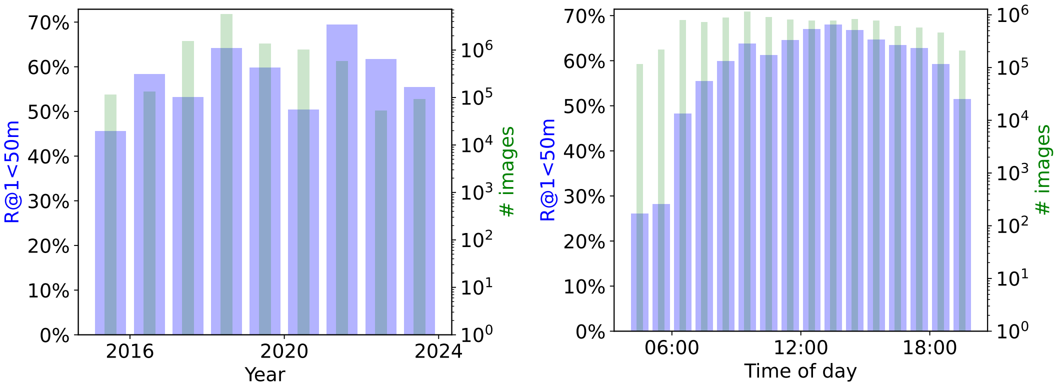


Fig. 4.16: Recall for photos in the Massachusetts test split captured in different years and during different times of the day. The aerial imagery in Massachusetts was taken during 2021 [86]. We only show buckets of the histograms with at least 50k samples.

structured and unstructured environments. If the top-1 predicted cell is incorrect, the probability distribution nevertheless often contains a significant peak at the correct location resulting in only few false positives cells that are scored higher than the true cell. Potential reasons for an incorrect prediction, aside from a challenging reference database, include extreme lighting conditions, image artifacts, a lateral camera perspective which is heavily underrepresented in the data on Mapillary, and ground-truth errors that are larger than 50m.

We further inspect the year and time of day that street-view images are captured as potential factors on the retrieval performance. Fig. 4.16 shows that the recall is highest for photos taken in the year 2021, *i.e.* when aerial imagery was captured over Massachusetts [86]. The performance drops when increasing the gap between the capture of query and reference images, likely due to the changing appearance of scenes over the timespan of several years[3]. However, even with a gap of five years, the model still correctly localizes about half of all query photos.

We additionally find that the performance is higher during the day, and drops at night (*cf*. Fig. 4.16, blue bars). The recall is 62.2% for photos captured between 6:00 and 18:00, and 46.6% for photos captured between 18:00 and 6:00. Potential reasons for the drop in performance include the change in illumination between the time periods, and a smaller amount of training data at night (*cf*. Fig. 4.16, green bars).

4.5.4 Ablation Studies

For the ablation studies, we choose a training setup with reduced computational cost (*cf*. Sec. 4.5.1) and evaluate on a smaller test split in Massachusetts (*cf*. Sec. 4.4.3).

[3] Photos from 2020 show a significant drop in performance. This correlates among others with the Covid pandemic and lockdowns which potentially changed the distribution of data uploaded to Mapillary in this year. However, we did not further investigate the causes of this outlier in this work.

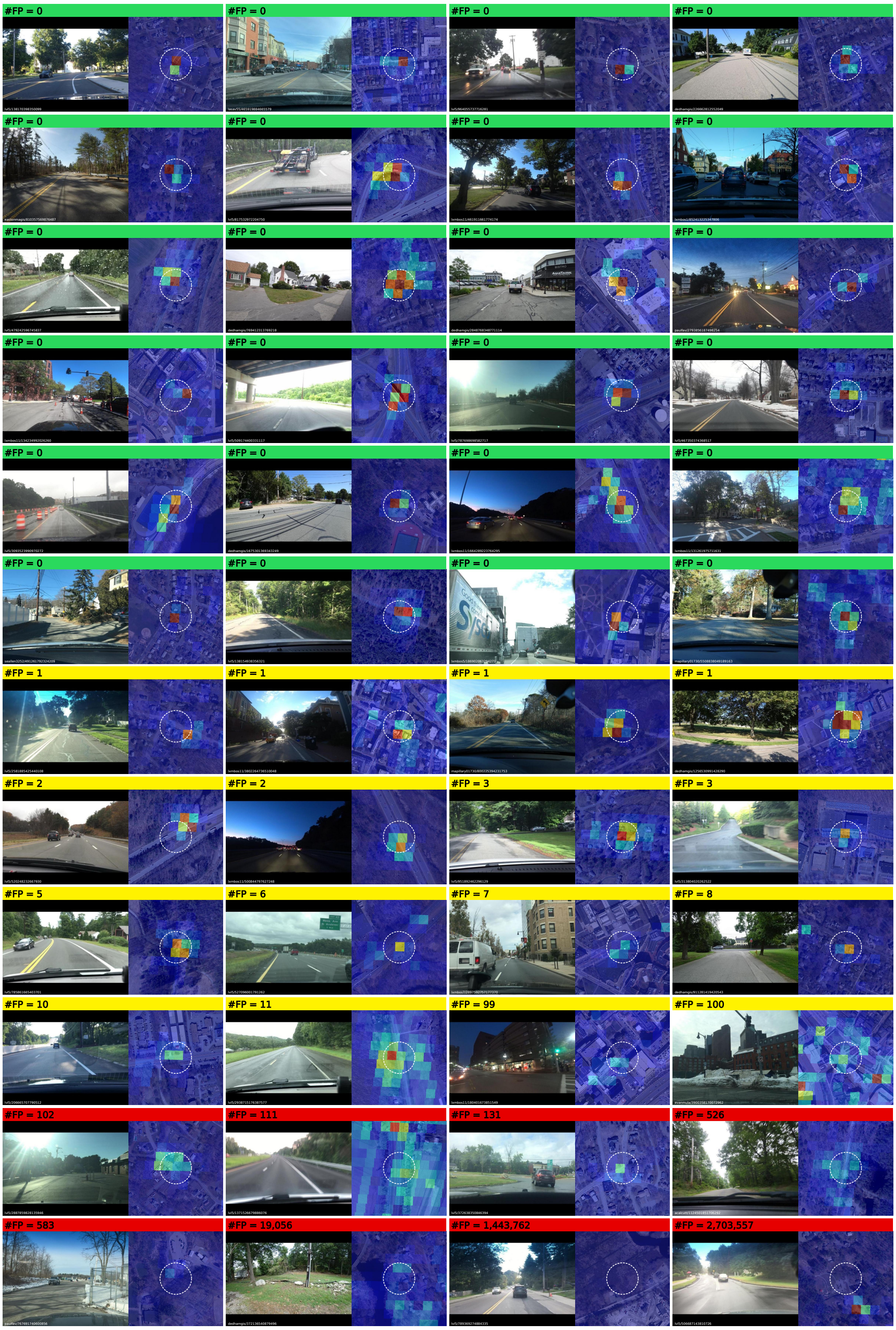


Fig. 4.17: Example predictions from our model on the Massachusetts test split covering an area of ~23000km^2. #FP indicates the number of false positives, *i.e.* cells in the search region that are scored higher by the model than cells in a 50m radius around the ground-truth location (*i.e.* dashed white circle).

Tab. 4.5: Evaluation of different choices of aerial images per cell on the ablation test split. We choose the resolution of images such that the single LoD and multiple LoD settings have the same number of input pixels and roughly the same computational cost.

Meters per pixel	Pixels	R@1<50m
0.12 [163]	512^2	33.4%
0.2	512^2	38.9%
0.4	512^2	40.6%
1.6	512^2	28.6%
0.2, 0.4, 0.8, 1.6	$4 \cdot 256^2$	48.3%
0.3, 0.6, 1.2, 2.4	$4 \cdot 256^2$	46.8%
0.4, 0.8, 1.6, 3.2	$4 \cdot 256^2$	46.1%

Tab. 4.6: Evaluation of different number of embedding dimensions on the ablation test split.

Embedding dimension	R@1<50m
512	45.7%
1024	48.3%
2048	50.0%
4096	50.4%

Tab. 4.7: Comparison of different pooling layers with the same encoder. SAFA and SMD are designed for a single LoD setting. We compare the single LoD and multiple LoD settings as in Tab. 4.5.

	R@1<50m	
Pooling layer	Multiple LOD	Single LOD
MHA (Ours)	48.3%	38.9%
Mean [30]	38.7%	29.3%
SAFA [108]	-	31.5%
SMD [165]	-	32.2%

Choice of aerial images Tab. 4.5 reports the recall of models trained with different choices of aerial images per search region cell (*cf*. Sec. 4.3.2). For fair comparison, we evaluate the single LoD setup with an input resolution of 512×512, and the multiple LoD setup with four images of size 256×256. In both cases, the model is thus given the same number of input pixels and has roughly the same computational cost.

The model trained with our multi-scale aerial imagery at the highest pixel resolution achieves the best recall of 48.3%. Increasing the size of the region around the cell that is covered by the images at the cost of a lower pixel resolution results in a slightly reduced recall.

The lowest performance is given by models that process only a single aerial image. However, even in this case the scale of the image is an important factor that has been overlooked by existing works. For instance, the VIGOR dataset [163] uses an aerial image at twice the side-length of the cell which corresponds to a pixel resolution of $0.12\frac{\text{m}}{\text{px}}$. Increasing this to $0.4\frac{\text{m}}{\text{px}}$ already improves the recall by 7.2% percentage points.

Embedding dimension Tab. 4.6 shows the recall of models that are trained with different numbers of embedding dimensions. A larger dimensionality of the

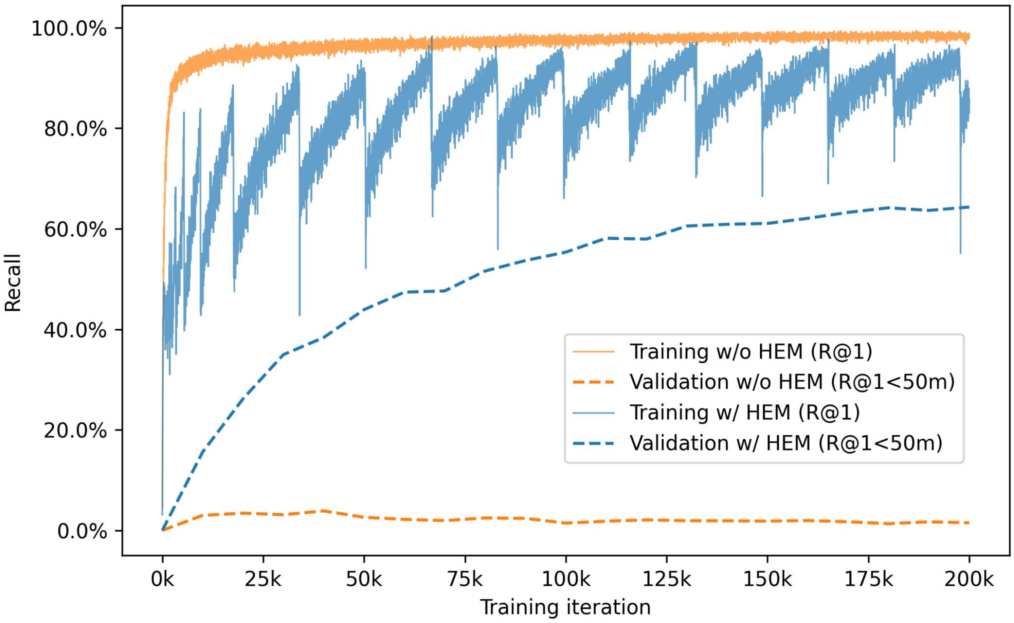


Fig. 4.18: Recall over the course of a training run. Without HEM, the per-batch recall quickly converges around 90% to 100%, and the validation recall stops improving. When applying our HEM strategy, the mean recall per training batch is reduced which results in stronger supervision even during later stages of training and prevents the validation score from stagnating early on. The training recall is depicted after smoothing with a Gaussian filter and $\sigma = 10$ iterations.

embedding space between 512 and 4096 increases the resulting recall, but results in a larger memory footprint of the reference database and slower retrieval. We choose 1024 dimensions as a trade-off between both factors.

Pooling layer Tab. 4.7 evaluates the performance that is achieved with different pooling layers. We vary only this block in the network and keep all other parameters unchanged. Our MHA layer in a multiple LoD setting yields the highest recall at 48.3%. Existing layers are designed for the single LoD setting; the best previous results are achieved by SMD at 32.2%. Even when applied only on a single aerial image, our MHA layer achieves a recall of 38.9%, highlighting the importance of the feature pooling layer when employing a CNN. While a mean-pooling layer is used by Deuser *et al.* [30] for a single LoD, we also evaluate it in a multiple LoD setting where it achieves a recall of 38.7%.

Hard Example Mining In the following paragraphs, we inspect the operation of our HEM strategy (*cf*. Sec. 4.3.5). Fig. 4.18 shows how the recall over individual training batches and over the test split changes over the course of a training run. Without HEM, the per-batch recall quickly converges around 90% to 100%, and the validation score stops improving. With HEM, the per-batch recall is reduced by increasing the hardness of samples, which provides sufficient supervision for the model even as its performance improves over the course of the training.

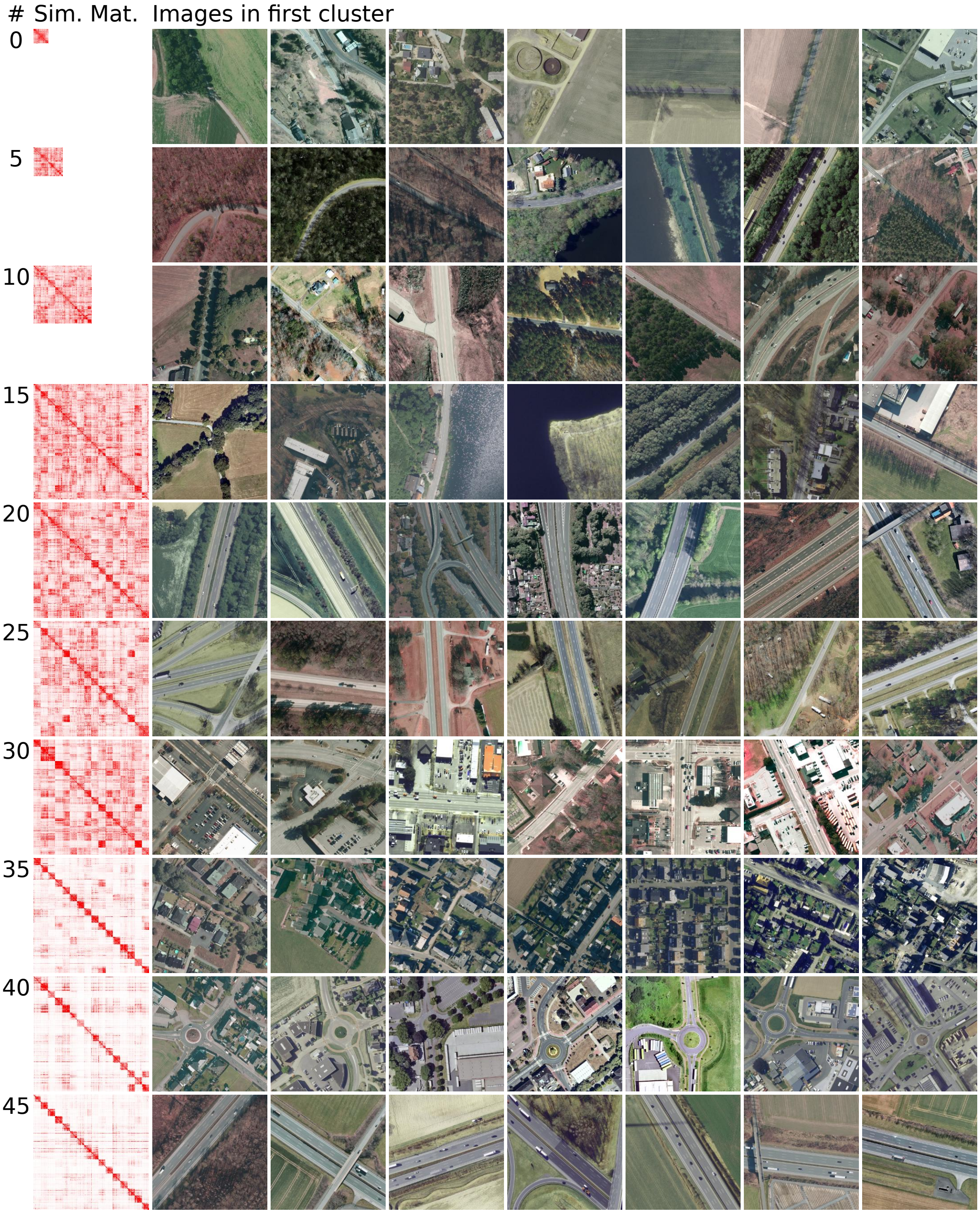


Fig. 4.19: Example clusters constructed by our HEM method over the course of a training run. Each row represents a single mining pool. (1) # indicates the sequence of mining pools during the training run. (2) The matrix shows the pairwise similarity between samples in the first 16 clusters of the pool, after applying the clustering method, as in Fig. 4.8. Early mining pools contain less than 16 clusters. As the training progresses, inter-cluster variance increases and intra-cluster variance decreases. (3) For each mining pool, we show several reference images from the first cluster. As the training progresses, images per cluster appear more similar and are harder for the model to discriminate.

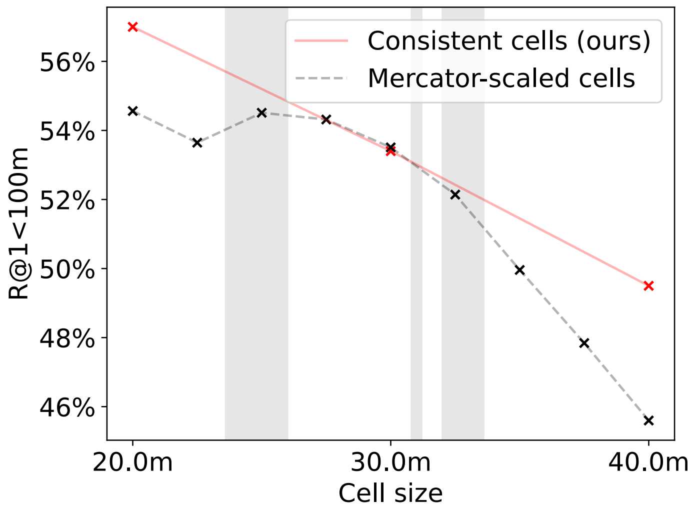


Fig. 4.20: Recall of models trained with our consistent cell layout and with a Mercator-based cell layout, evaluated on the ablation split. *Consistent cells:* We train and test three models with the cell sizes 20m, 30m and 40m to evaluate out consistent cell layout (solid red line). *Mercator-scaled cells:* To evaluate the Mercator-based layout, we train a model with a cell size of 40m at the equator. This results in varying cell sizes smaller than 40m across geographical regions. The gray highlighted regions indicate the cell sizes of samples in the training split and correspond to the latitudes of states in Germany and the US. We test the model over a range of cell sizes in the ablation split (dashed gray line) to investigate whether the inconsistent training cell sizes transfer to the given test cell size. For most cell sizes (*i.e.* latitudes), our consistent layout results in higher recall than the Mercator-based layout. The graph shows the recall with $r = 100$m to discount effects from the discretization of the search region into cells.

Fig. 4.19 shows example batches that are constructed by our HEM strategy over the course of a training run. Clusters initially contain reference images that depict many different types of scenes, and the problem of assigning a query photo to one of the references is easy to solve. As the training continues, the clusters represent progressively more difficult assignment problems due to the increasing size of the mining pool and performance of the model. For instance, row 40 in Fig. 4.19 depicts scenes that share a particular type of roundabout, and row 45 shows scenes with straight and similar sized highways.

Importantly, the creation of the training clusters does not require manual labeling of images or partitioning into different types of scenes. The clustering method is applied in an unsupervised manner and is based only on the learned embedding space of query and reference images.

Cell size Fig. 4.20 shows the localization accuracy that is achieved with different cell sizes with our consistent cell layout and a Mercator-based layout. We evaluate the recall with a radius $r = 100$m to discount effects from the discretization of the search region into cells.

Increasing the cell size requires the model to capture a larger region of possible camera locations in a single embedding representation and yields a lower recall. Doubling the cell size from 20m to 40m decreases the R@1<100m by roughly 7 percentage points. At the same time, it reduces the total number of cells that are required to cover the RoI, and thereby the size of the reference database and retrieval time. We choose a cell size of 30m as a trade-off between both factors.

Fig. 4.20 additionally shows the recall achieved with a Mercator-based cell layout. Here, the cell size corresponds to the latitude of the cell's geographical location (*cf*. Sec. 4.3.1), and the localization accuracy thus varies w.r.t. this latitude. We find that the Mercator-based layout achieves a comparable recall to our consistent layout only around the mean of all training latitudes, but drops significantly in geographical regions that deviate from this latitude. Cell sizes that are covered in the training data, but are too far from the mean latitude (gray regions in Fig. 4.20), already show a reduced recall. This highlights the importance of employing a search region layout with consistent cell sizes.

4.6 Summary

In this chapter, we revisit the problem of retrieval-based CVGL from the ground up and propose both a novel problem formulation as well as a comprehensive method that addresses various aspects of the task. Our work for the first time demonstrates the potential scalability of aerial imagery as reference data for VGL, as well as its real-world applicability by removing constraints such as a 360° FoV, north-aligned orientation or knowledge of camera parameters. Experimental evaluation shows that the method is able to localize 60.6% of all (non-panoramic) photos uploaded to the crowd-sourcing platform Mapillary in the state of Massachusetts to within 50m of their ground-truth location, without access to street-view data from this region for training or testing.

Our contributions are summarized as follows:

Search region layout We provide an analysis on how a RoI should be partitioned into geographical cells to factor in the curvature of the earth's surface. Based on this search region layout, we propose a novel problem formulation for retrieval-based CVGL as a *many-to-one* matching task that - unlike the *one-to-one* formulation - allows covering a RoI densely, and does not couple the size of aerial images and search region cells as in the *many-to-many* formulation.

Multiple LoD We propose utilizing aerial imagery at multiple LoD to provide more information to the embedding model per geographical cell and allow predicting discriminative features even over large RoI.

Model We propose a novel model architecture that utilizes MHA to aggregate local features predicted by a CNN over one or more images, and outperforms existing models in the field of CVGL both in a single LoD and multiple LoD setting.

HEM We propose a novel strategy for mining hard examples during training that shifts from optimizing the hardness of individual queries via a nearest neighbor approach to optimizing the hardness of individual batches via a clustering-based approach. This allows for a novel interpretation of HEM as simply approximating the use of a larger batch size in a contrastive training setting.

Speed We show that the retrieval time is reduced to less than 10ms by utilizing approximate nearest neighbor methods such as those implemented in FAISS [33].

Dataset We present a novel, large-scale dataset based on data from Mapillary and several orthophoto providers that is both larger and more diverse than existing datasets, and allows evaluating methods on a more than 100× larger RoI.

5 Pioneering Pose Estimation for Cross-view Geolocalization

5.1 Motivation

In recent years, autonomous driving technology has seen a significant surge in interest, driven by advances in the fields of machine learning and computer vision. Autonomous platforms are required to perceive the environment in order to navigate safely and efficiently. Two major paradigms have arisen that differ in how the internal model of the world is constructed.

Offline methods create parts of their model before deployment and use the live sensor data to update the model for instance with the locations of vehicles and pedestrians in the local environment. The offline model is represented by a High Definition (HD) map that is captured for instance using dedicated vehicles with lidar sensors. During deployment, the platform estimates its geo-pose by matching the live sensor data against the HD map [103]. Offline methods allow for a high localization accuracy of the system due to the availability of a high resolution, geo-registered model of the environment. However, due to the cost of creating and maintaining HD maps, they are typically only available in selected regions around the world.

Online methods rely entirely on the live sensor data, *e.g.* from lidar [153], camera [50] or both [79], to construct a model of the environment. This approach tends to be more scalable and avoids the need for expensive prior maps. However, the perception task in this setting is a more difficult problem since it does not rely on prior information about the environment.

Aerial imagery represents a potential trade-off between offline and online methods for autonomous driving. It provides the benefits of an offline map by giving access to prior information about the environment and facilitating sub-meter accurate localization, while retaining the scalability of online methods. Aerial images are available globally at high resolution and allow operating in regions that may not be prioritized for classical HD map creation.

5.2 Overview

In this chapter, we consider the problem of estimating the geo-pose of a platform by matching its camera readings against a single high resolution aerial image, *i.e.*

cross-view pose estimation. While this task has seen some interest in the research community, most works still rely on classical or hand-crafted feature matching approaches that are not able to robustly address the appearance change between street-view and aerial perspective without relying on additional input from range-scanners such as radar or lidar.

We present the first end-to-end learnable method for cross-view pose estimation that allows for application in a purely vision-based setup and achieves sub-meter accurate predictions. We integrate the model in a localization framework that enables long-term tracking of a platform's ego-trajectory. To evaluate our method, we gather several datasets from the autonomous driving sector as well as respective orthophotos, and refine their ground-truth geo-poses using automatic pseudo-label and data pruning approaches. Our contributions for the first time allow determining the geo-registered ego-pose and long-term ego-trajectory of a platform

1. using only aerial imagery as reference database (**RQ2**) without access to street-view data from the test region or test vehicle (**RQ3**),
2. in a purely vision-based manner without requiring range scanners such as lidar or radar (**RQ2**) or external signals such as GNSS,
3. across a range of environments such as urban and rural regions (**RQ3**), and
4. with sub-meter accuracy (**RQ2**).

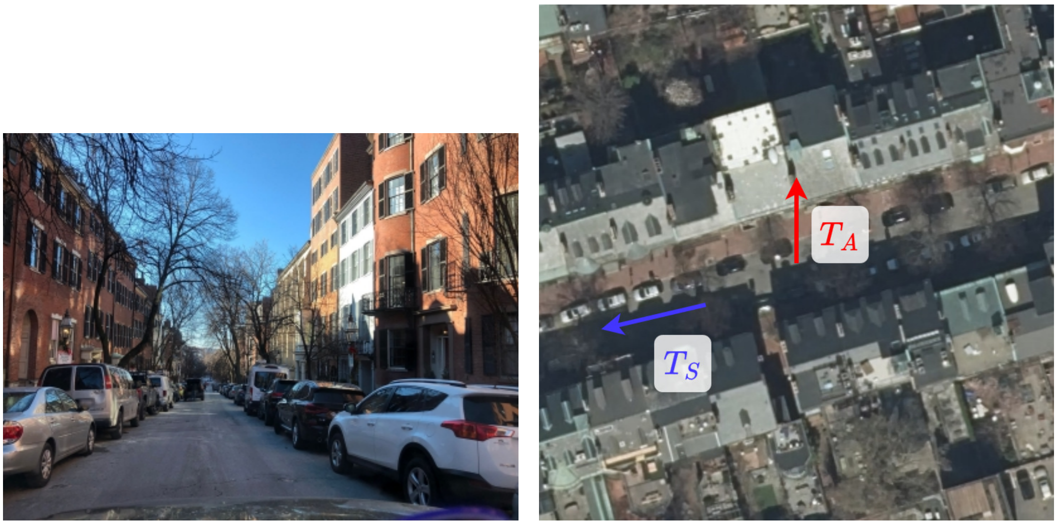
We describe our proposed method in Sec. 5.3, including details of the model architecture and loss function, and a novel filtering framework that allows integrating the model predictions over time to estimate the trajectory of a platform. In Sec. 5.4, we give an overview of the autonomous driving datasets and orthophotos that we use to train and evaluate our method, as well as the required preprocessing steps we applied to arrive at accurate ground-truth labels. In Sec. 5.5, we provide the results of our experimental evaluation of the pose and trajectory estimation.

This chapter is based on our publications in CVPR 2023 [38] and IROS 2022 [36].

5.3 Method

5.3.1 Problem Formulation

The cross-view pose estimation task is defined as follows. We consider a locally euclidean, metric coordinate system at some location on the earth's surface. All



(a) Camera image captured from the pose T_S.

(b) Aerial image centered on and aligned with the prior pose T_A.

Fig. 5.1: Cross-view pose estimation aims to predict the true pose of a platform T_S by matching its camera images against an aerial image centered on the prior pose T_A.

poses are given as rigid transformations $T = [R|t] \in \mathrm{SE}(2)$ with three DoF between local and reference coordinates[1] with $t \in \mathbb{R}^2$ and

$$R = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{5.1}$$

for an angle $\theta \in [0, 2\pi)$.

We aim to estimate the true pose $T_S \in \mathrm{SE}(2)$ of a platform (*e.g.* vehicle or hand-held device) that is located in proximity to a prior pose $T_A \in \mathrm{SE}(2)$ (*cf*. Fig. 5.1). The prior pose is given for instance via GNSS, cross-view retrieval or previous frames in a video. The maximum translation and orientation difference between T_S and T_A defines the extent of the search space. The true pose T_S is estimated by matching one or more camera images from the platform against an aerial image that is centered on and aligned with the prior pose T_A.

All sensor data from the platform are captured synchronously up to a negligible delay. The extrinsic calibration of the sensors is known which allows transforming all measurements into the common reference frame of the platform.

5.3.2 Model Architecture

We propose a novel model architecture that takes as input one or more images captured synchronously from the sensor platform, as well as an aerial image centered

[1] SE(2) is the two-dimensional special euclidean group, *i.e*. the set of rigid transformations in a two-dimensional space.

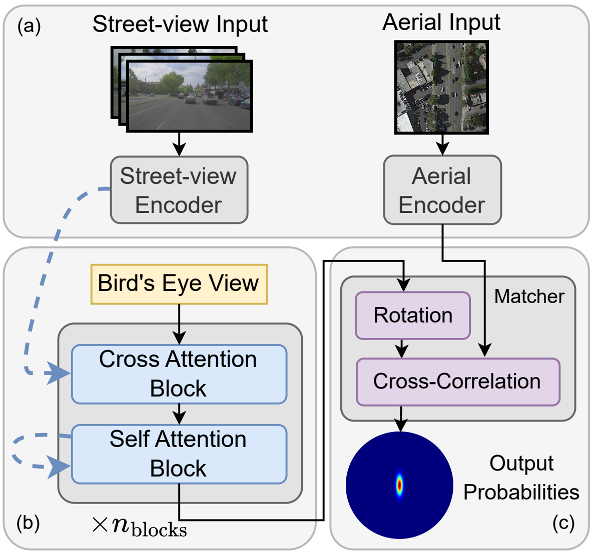


Fig. 5.2: Overview of the model architecture. (a) A vision backbone extracts high-level features from the input images. (b) A cross-attention mechanism is used to project the features from street-view to BEV representation. (c) The final BEV and aerial features are matched to predict a probability distribution over possible platform poses.

on the prior pose, and predicts a probability distribution over possible platform poses (*cf*. Fig. 5.2). The model first extracts high-level features from the input images using ConvNeXt-based encoder networks [80]. The features for the street-view images are projected into a joint Bird's Eye View (BEV) representation using a cross-attention mechanism [124]. The BEV map and aerial features are compared densely to determine the probability distribution over possible poses. The entire model is trained end-to-end under the supervision of a pose estimation loss. The details are described below.

Feature extraction In a first step, the street-view and aerial images are processed by vision encoders that extract high-level feature maps from the low-level color input at a stride of s_0, *i.e*. with $\frac{1}{s_0}$ of the original spatial resolution (*cf*. Fig. 5.3 and Fig. 5.2a). We use a similar architecture, but different weights for the street-view and aerial domains. Street-view images from different onboard cameras are encoded with the same network and weights.

The encoder network consists of a ConvNeXt backbone [80] pretrained on ImageNet [29] that extracts feature maps at varying resolutions, a context pooling module that provides a global receptive field, and a decoder that maps multiple intermediate feature maps onto the final output features.

ConvNeXt applies a total of four stages of layers to the input image that progressively downsample the resolution and yield more abstract features. The outputs of the four stages are used as intermediate feature maps with resolutions reduced by the strides 4, 8, 16 and 32. Given the local receptive field of ConvNeXt, we add a simple

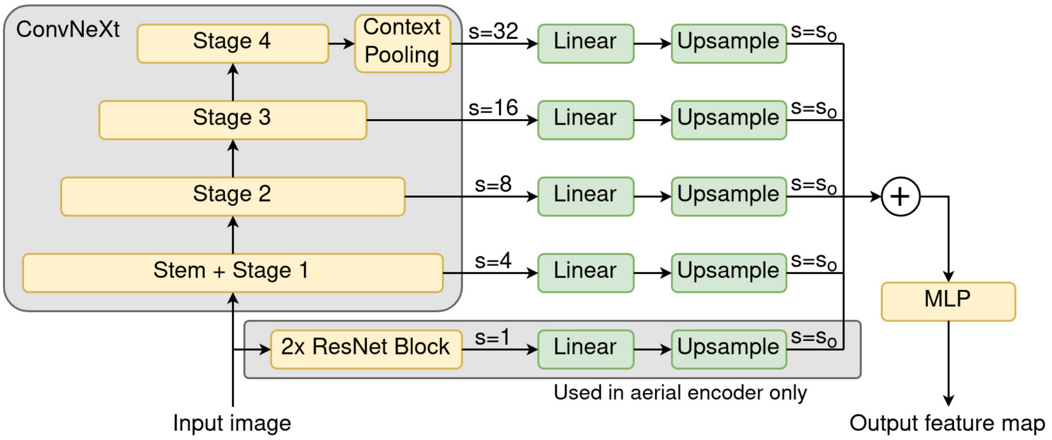


Fig. 5.3: Overview of the network used to extract a high-level feature map from an input image. A ConvNeXt backbone [80] encodes the input image to feature maps at different strides s. We use a context pooling module based on a global average at the last feature map to enhance the model with a global receptive field. All feature maps are upsampled to stride s_0, added and mapped to c channels via a small MLP to yield the final output feature map. In the aerial domain, an additional path consisting of two ResNet blocks [52] is used that retains the highest resolution with stride 1.

context pooling module to the last feature map as follows. We compute the spatial average of the features, process it with a small MLP and append it along the channel dimension to all spatial locations in the original feature map [25, 157].

All intermediate feature maps are mapped to the same number of channels, upsampled to the same stride s_0 and added. The resulting feature map thus combines low-level features with high spatial resolution from the earlier stages of ConvNeXt, and high-level features with low spatial resolution from the later stages. In a last step, the features are mapped onto the desired number of channels via a small per-pixel MLP.

In the street-view domain, we use a final output stride of $s_0 = 4$ to reduce the memory requirement of the resulting feature map. In the aerial domain however, the spatial resolution of the features is particularly important for the localization accuracy. We therefore use the highest resolution stride of $s_0 = 1$. Since the intermediate feature maps are given at a minimum stride of 4, we encode an additional low-level feature map from the input image at a stride of 1 using two ResNet blocks [52] and add it to the list of intermediate feature maps.

We define the feature map of the i-th street-view camera as $F_{Si} \in \mathbb{R}^{h_{Si} \times w_{Si} \times c_S}$ with stride 4, and the feature map of the aerial image as $F_A \in \mathbb{R}^{h_A \times h_A \times c_A}$ with stride 1.

Perspective View to Bird's Eye View In the second step, the features from all street-view cameras are transformed into a joint BEV representation (*cf*. Fig. 5.2b). The BEV is given as a two-dimensional feature map $F_B \in \mathbb{R}^{h_B \times h_B \times c_B}$ with a side-length of h_B and c_B channels that is centered on the platform pose and depicts the

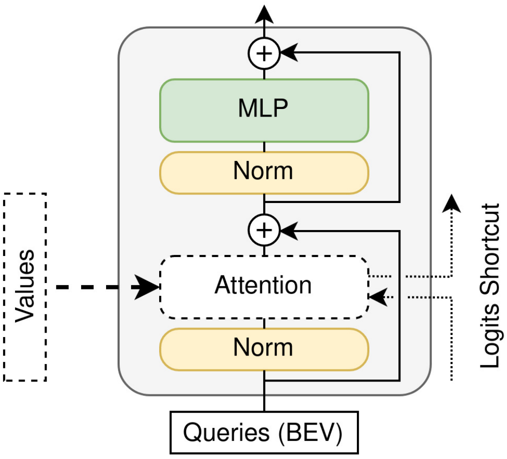


Fig. 5.4: General structure of the transformer block [124]. A set of queries in BEV gathers information from a set of values, followed by a per-pixel MLP. In the case of cross-attention, the values are the features extracted from the street-view images. In the case of self-attention, the queries and values are determined from the same input BEV and allow for communication across its spatial locations.

surrounding environment. We consider only cells in F_B with a distance of at most $0.5h_B$ to the platform as valid and store the resulting mask as $M \in \{0,1\}^{h_B \times h_B}$.

The BEV map F_B is constructed from the street-view features F_{Si} by iteratively applying cross-attention and self-attention layers as depicted in Fig. 5.2b. We start with a learnable map of BEV features that represents the prior belief of the network about the vehicle's environment. Each subsequent application of a cross-attention block refines the BEV map by gathering information from the street-view features. Each self-attention block refines the BEV map by allowing different cells to communicate with each other. The final BEV map is upsampled via a bilinear interpolation to match the higher resolution of the aerial feature map F_A, and masked according to M.

The general structure of the self- and cross-attention blocks follows the transformer architecture [124] and is shown in Fig. 5.4. The following paragraphs provide a more detailed description.

Cross-attention In the cross-attention block, the BEV features are defined as queries that gather information from the feature maps of the street-view images. Computing the full attention from each cell in the BEV to all street-view features would result in large memory and computational overhead. Instead, we encode a geometric bias in the attention block that significantly reduces the complexity of the operation.

Each cell in the BEV corresponds to a vertical column in three-dimensional space around the platform. We define a cell to attend to only those parts of the street-view images that the vertical column is projected onto, and ignore information from

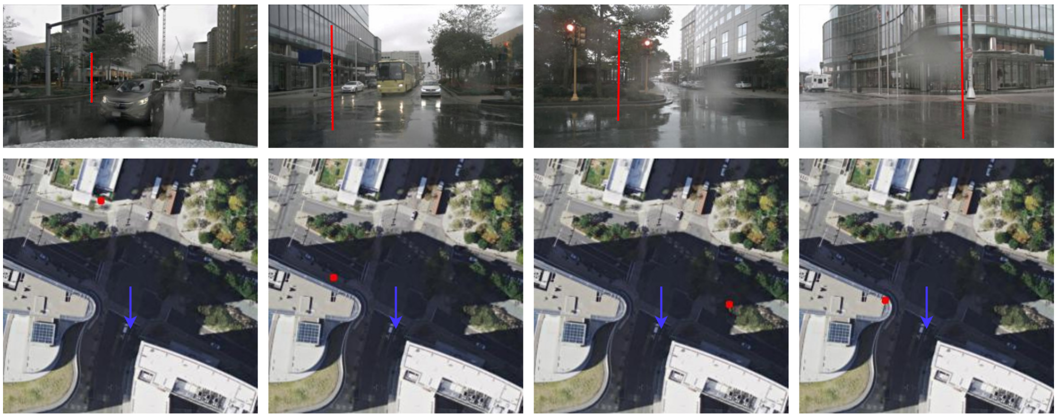


Fig. 5.5: *Top row:* Four surround camera images captured from a vehicle in the Nuscenes [18] dataset, and four pillars of points (red lines) that are constructed for four BEV cells. The pillars are created between the heights of $h_{\text{min}} = -5\text{m}$ and $h_{\text{max}} = 10\text{m}$ relative to the ground level. *Bottom row:* Aerial images that depict the vehicle environment from a BEV perspective. The red dots mark the locations of the BEV cells for which point pillars are shown. The blue arrows show the location and heading of the vehicle. Map Data: Bing Maps © 2022 TomTom, © Vexcel Imaging.

any other location. For instance, a BEV cell positioned in front of the vehicle does not receive information from street-view cameras that face the sides or back of the platform.

The attention operation is implemented as follows. For each query cell, we generate a pillar of z points in its center that are spaced uniformly between the heights h_{min} and h_{max} [73]. The points are projected from BEV coordinates into the camera coordinate systems using the cameras' extrinsic and intrinsic parameters. We gather features from the street-view feature maps at the locations of the projected points via bilinear interpolation, and use them as values for the respective BEV query. Fig. 5.5 shows examples of BEV cells and corresponding pillars in the street-view images.

A pillar represents a sparse set of positions in the street-view feature map that the BEV query gathers information from. To allow the network to more flexibly choose the locations from which values are sampled, we add deformable offsets to all points before applying the bilinear interpolation as follows [164].

Given a BEV cell and its feature representation, we predict z two-dimensional offsets $\Delta p_j \in \mathbb{R}^2$ via a small MLP with $j \in \{1, ..., z\}$. Given the pixel location $p_{ij} \in \mathbb{R}^2$ that corresponds to the j-th point in the pillar on the i-th camera, the corresponding feature f_{ij} is sampled via bilinear interpolation as

$$f_{ij} = F_{Si}(\frac{p_{ij} + \Delta p_j}{s_S}) \tag{5.2}$$

where s_S represents the stride of the feature maps F_{Si}.

We compute the cross-attention from each query to the values that are gathered according to the above description. The values represent the features predicted from the street-view images for the location of the BEV cell at z distinct heights. To simplify the computation, we use the same z keys in the attention operation of all queries. The keys are defined as learnable parameters for the z predefined heights between h_{min} and h_{max}. The network is thus conditioned to estimate the height distribution of objects in the scene, such that the attention operation is able to gather features from the correct locations in the street-view images.

Finally, we add skip connections to ease the iterative estimation of the BEV features F_B as follows. We add the deformable offsets used in a cross-attention block onto the offsets of the next block, such that each block learns to refine the existing offsets rather than predict entirely new offsets. The attention logits per query are forwarded to the next cross-attention block via an analogous skip connection (*cf*. Fig. 5.4) [53].

Self-attention The self-attention block allows different cells in the BEV map to communicate which each other and is applied in an alternating manner with the cross-attention blocks. Computing the full self-attention between all cells in the BEV would result in large memory and computational overhead due to the quadratic cost w.r.t. the size of the BEV map, and quartic cost w.r.t. the its side-length h_B. Instead, we use a single block of the SegFormer architecture [148] which reduces the complexity of the operation as follows.

Each cell in the BEV map is defined as a query in the self-attention operation. The keys and values area determined by first downsampling the BEV map with a stride of s_R. Each key-value pair those contains information from s_R^2 cells and thereby reduces the complexity of the attention operation by a factor of s_R^2. The MLP component is additionally extended with a 3×3 convolution to mimic the use of a positional encoding, and invalid cells according to the mask M are ignored.

Matching The features F_A and F_B are given as top-down, orthogonal maps that are determined from the aerial image and the street-view images and centered on T_A and T_S, respectively. The relative transformation $T_{A \to S}$ is determined by matching F_A and F_B as follows, and allows computing the platform pose $T_S = T_{A \to S} \cdot T_A$.

We consider a set of hypotheses $\mathcal{H}$ for $T_{A \to S}$. For each hypothesis $h \in \mathcal{H}$, we transform F_A into the coordinate system T_S using the transformation T_h, resulting in the transformed $F_A^{(h)}$ with the same dimensions as F_B. The logit (*i.e.* unnormalized log-probability) of h is determined as the scaled inner product of $F_A^{(h)}$ and F_B under the transformation T_h:

$$\ell(h) = \frac{\langle F_A^{(h)}, F_B \rangle}{\tau} \text{ with } \tau = \sqrt{\langle M, M \rangle \cdot c_A} \tag{5.3}$$

We normalize the variance of the inner product by scaling with the inverse square root of the number of elements in the feature representation as proposed by Vaswani *et al.* [124]. The value $\langle M, M \rangle$ is the number of spatial locations, and c_A the number of channels per location.

A softmax operation is used to convert logits to probabilities with a sum of one:

$$P(h) = \frac{\exp \ell(h)}{\sum_{h' \in \mathcal{H}} \exp \ell(h')} \tag{5.4}$$

Efficient matching Computing the logit of each hypothesis separately results in a large computational overhead when considering many hypotheses. Instead, we design the set $\mathcal{H}$ to allow for multiple hypotheses to be evaluated jointly using an efficient cross-correlation operation as follows. We define

$$\mathcal{H} = \bigcup_{\theta \in \Theta} \mathcal{H}_\theta \tag{5.5}$$

as the union of disjunct subsets $\mathcal{H}_\theta$. All poses in $\mathcal{H}_\theta$ have the same rotation $\theta \in \Theta \subset \mathbb{R}$. The translations in $\mathcal{H}_\theta$ are sampled in a regular grid that mirrors the pixel grid of the aerial image. This allows for all hypotheses in $\mathcal{H}_\theta$ to be evaluated jointly by rotating F_A once, and computing the cross-correlation between the rotated $F_A^{(\theta)}$ and F_B:

$$\ell(\mathcal{H}_\theta) = \frac{1}{\tau} F_A^{(\theta)} * F_B \tag{5.6}$$

We use the Convolution Theorem [141] to reduce the cross-correlation to a simple element-wise multiplication in the Fourier domain:

$$\ell(\mathcal{H}_\theta) = \frac{1}{\tau} F_A^{(\theta)} * F_B = \frac{1}{\tau} \mathcal{F}^{-1}(\mathcal{F}(F_A^{(\theta)}) \cdot \overline{\mathcal{F}(F_B)}) \tag{5.7}$$

Deep learning frameworks such as Jax [16] and PyTorch [98] provide differentiable implementations of the Fourier transformation $\mathcal{F}$ which allows utilizing the optimization both during training and testing stages. We find that computing the cross-correlation without this optimization results in a slowdown of more than $\times 100$ when using Jax on the Central Processing Unit (CPU), and is not feasible when using Jax on GPU due to the unmanageable memory requirement of employing a very large kernel size.

The angles in Θ are chosen in regular intervals over the range of orientations that are considered in a given scenario. The maximum translational offset is chosen such that the BEV map still entirely fits onto the aerial feature map.

5.3.3 Loss Function

Each training sample contains street-view images that are captured synchronously from the platform at T_S, the extrinsic and intrinsic parameters of the respective cameras, and an aerial image that is shifted w.r.t. T_S by a randomly sampled $T_{A \to S}$ and represents the prior pose. The street-view images are fed to the model to predict features for the matching step. The camera parameters are used to project the features into BEV where they are matched with the aerial image. The final probability distribution P predicted for $T_{A \to S}$ is supervised by the loss as follows.

We define a target distribution P_{true} over the domain of P as a normal distribution with the mean $T_{A \to S}$ and standard deviations of σ_t and σ_θ for the translation and rotation:

$$P_{\text{true}}(h) = \mathcal{N}(T_{A \to S}, \begin{pmatrix} \sigma_t^2 \\ \sigma_\theta^2 \end{pmatrix}) \tag{5.8}$$

We apply a cross-entropy loss between P and P_{true} to supervise the model to align the predicted probability distribution with the target distribution. The loss is computed as follows:

$$L = -\sum_{h \in \mathcal{H}} P_{\text{true}}(h) \log P(h) \tag{5.9}$$

Defining P_{true} as a soft normal distribution rather than a one-hot distribution over pose hypotheses has several advantages:

- Increasing the standard deviations σ_t and σ_θ allows compensating for noise in the ground-truth pose of the platform by reducing the penalty for small prediction errors.
- A one-hot distribution would require the ground-truth pose to coincide exactly with one of the pose hypotheses $h \in \mathcal{H}$. The soft distribution allows training with ground-truth poses that lie between the evenly spaced hypotheses.
- The soft distribution acts as a method for label smoothing which among others prevents the model predictions from becoming over-confident [91].

5.3.4 Trajectory Estimation

The model described in previous sections allows performing cross-view pose estimation of a platform given its sensor data at some point in time. Next, we consider the problem of how to integrate the predictions in a tracking framework that allows predicting the platform's trajectory over time.

We estimate the state of the platform as a six-tuple of values following the Constant Turn-rate and Acceleration (CTRA) motion model [115]:

$$x = \begin{bmatrix} p_1 \\ p_2 \\ v \\ a \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{5.10}$$

The vector $p \in \mathbb{R}^2$ is the platform's location, v and a are the velocity and acceleration in the forward direction, θ and $\dot{\theta}$ are the orientation and angular velocity in $\mathrm{SE}(2)$.

Bayes filter We make use of the Markov assumption and keep track of a probability distribution over the state using a Bayes filter [68]. The filter starts with a prior state estimate $P(x_0)$ and recursively applies the **predict** and **update** steps to track the state over time and integrate measurements z_t:

1. In the **predict** step, the filter's state estimate at time $t-1$ is propagated to the future time step t following the motion model, without integrating additional measurements. To do this, the CTRA motion model assumes that the values of $\dot{\theta}$ and a are constant and adjusts the other state estimates accordingly.

   The equation computed in the predict step is as follows:

   $$P(x_t|z_{1:t-1}) = \int_{x_{t-1}} P(x_t|x_{t-1}) \cdot P(x_{t-1}|z_{1:t-1}) dx_{t-1} \tag{5.11}$$

   The term $P(x_{t-1}|z_{1:t-1})$ represents the state estimate at time $t-1$ and is based on all measurements z up to that time. The predict step computes the next state estimate $P(x_t|z_{1:t-1})$ without integrating any additional measurements. The propagation is based on the process model $P(x_t|x_{t-1})$, *i.e.* the CTRA motion model.

2. In the **update** step, a measurement z_t given at time t is integrated into the state estimate of the filter. The measurement is the output of the pose estimation model and comprises values for p_1, p_2 and θ.

   The equation computed in the update step is as follows:

   $$P(x_t|z_{1:t}) \propto P(z_t|x_t) \cdot P(x_t|z_{1:t-1}) \tag{5.12}$$

   The term $P(x_t|z_{1:t-1})$ represents the state estimate at time t before integrating the measurement z_t. It is multiplied according to Bayes' theorem with the likelihood $P(z_t|x_t)$, *i.e.* the output of the pose estimation model (*cf*. Eq. (5.4)), resulting in the updated state estimate $P(x_t|z_{1:t})$.

Kalman filter The Kalman filter represents a particular type of Bayes filter that assumes that all distributions are Gaussian (*i.e.* uni-modal) and the motion and observations models are linear [60]. This allows Eq. (5.11) and Eq. (5.12) to be written as simple matrix multiplications.

Most works in cross-view trajectory estimation rely on a particle filter instead (*cf.* Sec. 3.5), which allows modeling complex probability distributions that result from noisy, non-Gaussian pose estimations. We propose a novel method to transform the multi-modal output of the pose estimation into a Gaussian representation, which allows using the Kalman filter for trajectory tracking even with non-Gaussian pose predictions. In addition to providing a uni-modal trajectory estimate, it also facilitates fast, real-time performance due to the lower computational cost of the Kalman filter compared to particle filters[2] [112].

Applying the Kalman filter to our problem setting requires two modifications. Firstly, the CTRA motion model does not satisfy the linearity requirement of the Kalman filter. We therefore instead employ an Extended Kalman Filter (EKF) that linearizes the motion model around the current state estimate. Secondly, the output of the pose estimation model $P(z_t|x_t)$ is not given as a normal distribution, but as a multi-modal probability distribution that represents discrete hypotheses sampled over the pose domain. The following paragraphs discuss how we transform the model output into a normal distribution, such that the Kalman filter is able to process it.

Adapting the model output The output of the model is given as a probability distribution $P(h)$ over pose hypotheses $h \in \mathcal{H}$ and converted to a normal distribution $\mathcal{N}(\mu, \Sigma)$ as follows.

A naive approach, called $\mathcal{N}_A$, would directly compute μ and Σ as the statistical mean and covariance of the discrete hypotheses represented in P:

$$\mu_A = \mathrm{E}(P) \qquad \text{and} \qquad \Sigma_A = \mathrm{Var}(P) \tag{5.13}$$

In general, P does not follow a normal distribution and might have multiple local maxima across the search space[3]. Such a multi-modal prediction results in an estimate of μ that potentially lies in between the maxima, and a large uncertainty in Σ that reflects the spread of the peaks. Measurements with large uncertainty are downweighted by the Kalman filter and result only in small updates to the state estimate[4]. Model prediction's with multiple peaks are thus largely ignored, and the filter falls back to a prior trajectory that is computed in the predict step according to the motion model.

[2] See Fig. 15.7 on p. 480 in *Optimal State Estimation* [112]. Simon *et al.*: "it is the computational burden of the particle filter that is its primary obstacle" (p. 461)

[3] Fig. 1.5 shows an example of this phenomenon.

[4] A normal distribution with large uncertainty is close to a uniform distribution. Inserting a uniform distribution for $P(z_t|x_t)$ in the update step (*i.e.* Eq. (5.12)) results in identical input and output state estimates.

We propose to transform the model output into a normal distribution by relying on the prior distribution over poses P_{prior} that is provided by the Kalman filter's current state estimate. Instead of computing the statistical mean and covariance across all hypothesis in the search space equally, we weight hypotheses based on their prior probability. Local maxima in P that are too far from the prior and have a low P_{prior} thus have little effect on the resulting μ and Σ. We call this approach $\mathcal{N}_B$:

$$\mu_B = \mathrm{E}(P \cdot P_{\text{prior}}) \quad \text{and} \quad \Sigma_B = \mathrm{Var}(P \cdot P_{\text{prior}}) \tag{5.14}$$

This addresses the problem of multi-modal model predictions. However, multiplying P with P_{prior} also reduces the uncertainty of Σ_B according to the confidence encoded in P_{prior}, and thus leads to a self-reinforcing decrease in uncertainty. For instance, a distribution P_{prior} with a single peak and high confidence results in a low uncertainty in Σ_B, which is then used in the update step of the Kalman filter to reduce the uncertainty of the next P_{prior} even further.

To counteract this effect, we increase the uncertainty of Σ by the amount that was reduced through the multiplication with P_{prior}, and call the strategy $\mathcal{N}_C$:

$$\mu_C = \mathrm{E}(P \cdot P_{\text{prior}}) \tag{5.15}$$

$$\Sigma_C = \mathrm{Var}(P) \cdot (\mathrm{Var}(P) - \mathrm{Var}(P \cdot P_{\text{prior}}))^{-1} \cdot \mathrm{Var}(P \cdot P_{\text{prior}}) \tag{5.16}$$

This strategy acts as a soft windowing method that focuses the approximation of μ and Σ on a region of hypotheses around the prior proportional in size to the prior's uncertainty, but does not self-reinforce its confidence.

Additional input The Kalman filter allows integrating measurements from other sensors in addition to those provided by the pose estimation model. In our experiments, we use the acceleration and angular velocity measurements provided by an Inertial Measurement Unit (IMU) which increases the smoothness of the local trajectory. Other scenarios include noisy GNSS measurements that are refined via our cross-view pose estimation for more accurate trajectory estimation over time.

5.4 Data

We train and evaluate our cross-view pose and trajectory estimation method on datasets from the autonomous driving sector. We use aerial imagery from local state providers as in Chapter 4 where possible. However, many driving datasets are captured in states that do not provide open data access to aerial imagery. We thus additionally use orthophotos from Google Maps and Bing Maps, which offer global availability at high resolution.

Tab. 5.1: Street-view datasets used to evaluate our cross-view pose estimation method. The data contain the vehicle's geo-pose, camera images and lidar scans as well as intrinsic and extrinsic parameters. Samples are divided into disjoint cells with size 100m × 100m to measure aerial coverage (*cf*. Sec. 4.3.1). SD: Average scene duration in seconds.

Dataset	Region	Year	Scenes	Frames ($\times 10^3$)	SD (sec)	Cams	Cells
Argoverse V1 [22]	Miami	$\leq$ 2019	53	12	22	9	71
	Pittsburgh	$\leq$ 2019	60	10	17	9	55
Argoverse V2 [143]	Austin	$\leq$ 2021	111	48	43	7	296
	Detroit	$\leq$ 2021	256	91	36	7	569
	Miami	$\leq$ 2021	703	245	34	7	811
	Palo Alto	$\leq$ 2021	43	136	34	7	157
	Pittsburgh	$\leq$ 2021	668	228	34	7	557
	Washington	$\leq$ 2021	262	90	34	7	553
Ford AV [3]	Detroit	2017	18	136	811	6-7	983
KITTI-360 [74]	Karlsruhe	2013	9	76	877	3	609
Lyft L5 [56]	Palo Alto	2019	398	50	25	6	88
Nuscenes [18]	Boston	2018	467	19	20	6	174
Pandaset [147]	Palo Alto	2019	35	3	8	6	87
	San Francisco	2019	65	5	8	6	93
Total (Σ)	–	–	3148	1149	37	–	–

Most datasets capture the geo-poses of vehicles using GNSS and optional post-processing steps. However, we find that the trajectories are often inaccurate and deviate from the correct locations on the aerial images by several meters. We implement a pseudo-label approach to refine the ground-truth, and a data pruning approach to filter out invalid data samples, *e.g*. where the vehicle is traveling through a tunnel.

The following sections provide an overview of the data and preprocessing steps.

5.4.1 Street-view Data

The street-view data is taken from several datasets that are widely used in the field of autonomous driving, including Argoverse V1 and V2 [22, 143], Ford AV [3], KITTI-360 [74], Lyft L5 [56], Nuscenes [18], and Pandaset [147] (*cf*. Tab. 5.1). The datasets include a full or partial surround view of the vehicle with cameras and lidar sensors, continuous trajectories (*i.e*. scenes) with lengths between several seconds and minutes, and are spread across several cities in Germany and the US. In total, the data comprise 1.15M samples; each sample contains the surround camera images and lidar points at a given point in time, the ground-truth pose and intrinsic and extrinsic parameters. We resize and pad all images to a resolution of 320×240.

Data samples are captured with a high frame-rate, *e.g*. 10 Hz in the Ford AV dataset [3]. Subsequent frames in a scene are thus located in close proximity to each other. Furthermore, the same route is often captured in multiple drives at different points in time. Each dataset thus contains a large diversity of street-view samples, but provides a much smaller coverage of aerial imagery. For instance, the Lyft L5 dataset consists of 50k frames spread over just 88 geographical cells of size 100m×100m. We combine multiple datasets to alleviate this problem and provide better generalization across aerial imagery, types of vehicles and camera setups.

5.4.2 Aerial Data

We collect aerial imagery for the street-view datasets from the respective states' open data providers (*cf*. Sec. 4.4.1) if possible. This includes the Nuscenes dataset which is captured in Boston, Massachusetts [86]; and parts of the Argoverse V2 dataset captured in Austin, Texas [113] and Washington D.C. [95]. However, the majority of scenes are captured in states without open data policies for the respective orthophotos. We thus download additional aerial imagery for all locations from Google Maps and Bing Maps, which provide orthophotos globally and at high resolution.

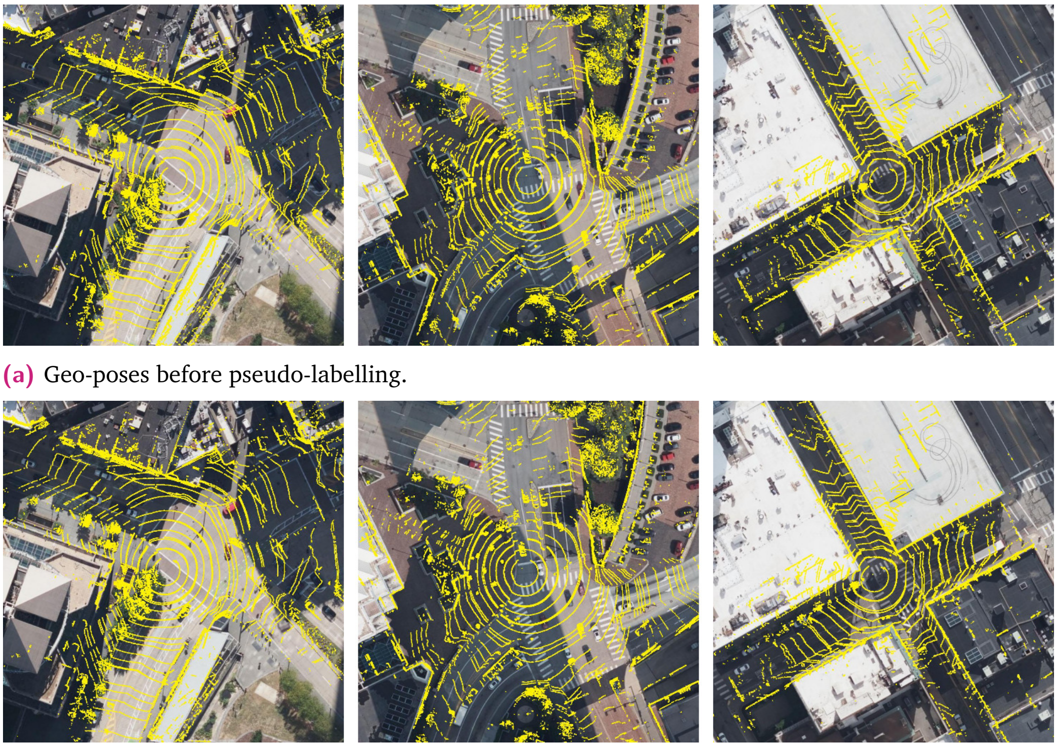
Unlike in cross-view retrieval, aerial imagery is only required for the locations of the captured scenes and does not have to be downloaded densely over a RoI. We use the same online sampler as in our retrieval method (*cf*. Sec. 4.4.1) to retrieve aerial images with a consistent pixel resolution and centered on the prior poses T_A. A consistent resolution is particularly important in cross-view pose estimation, since the BEV is implicitly constructed by the model to match the scale of the paired aerial image. Inconsistent resolutions during training would prohibit the model from predicting the BEV with a known scale. Inconsistent resolutions between training and test splits would result in a domain gap.

5.4.3 Pseudo-labels

Similar to previous works [135, 107], we find that geo-poses included in the street-view datasets are often inaccurate by up to several meters and do not align with the geo-registered aerial imagery. Poses are typically measured using GNSS and optional post-processing steps. Fig. 5.6a shows examples of lidar point clouds projected into an aerial image centered on the respective geo-pose. Visible edges in the point cloud that are reflected from vertical structures in the scene do not align with the matching structures in the aerial image, indicating an error in the underlying geo-pose.

We propose a pseudo-labelling method to refine the ground-truth poses based on the predictions of a lidar-based model. The refined poses are shown in Fig. 5.6b; lidar points and aerial image are in alignment. The method comprises several steps and is described in detail in the following paragraphs.

Step 1 We manually label a subset of the data using a custom tool that allows shifting and rotating the lidar point cloud on an aerial image until they are in alignment. The new geo-pose for the respective frame is computed by applying the



(a) Geo-poses before pseudo-labelling.

(b) Geo-poses after pseudo-labelling.

Fig. 5.6: Visualization of the geo-pose error before and after applying our pseudo-label method. The images show orthophotos centered on the vehicle pose at some point in time, as well as the respective lidar point cloud projected into BEV. The alignment between edges in the lidar points and vertical structures on the aerial image allows qualitatively approximating the geo-pose error. Vehicle data: Argoverse V2 [143]. Map Data: Bing Maps ©2022 TomTom, © Vexcel Imaging.

relative transformation determined in the tool to the original geo-pose. We label only a small, sparse subset of frames per trajectory and interpolate the geo-poses of the remaining frames based on the relative transformation between frames provided in the datasets.

Step 2 We train a pseudo-labelling model on the manually labeled data. The model follows a similar architecture as our main model. However, we replace the expensive transformer-based PV2BEV module with a simple projection utilizing a synchronously captured lidar point cloud (*cf*. Fig. 5.7). The lidar scan provides an explicit model of the three-dimensional environment which greatly simplifies the matching task for the model. It consequently requires less data to train and still yields accurate localization results. The lidar points are thus used as a type of privileged information that is available during training, but not during testing.

Step 3 We predict geo-poses for all samples in the full dataset using the pseudo-labelling model that was trained on a subset of the data.

Step 4 We refine the predicted pseudo-labels using a least-squares approach. For each trajectory, we build a pose graph [47] and insert both the predicted geo-poses

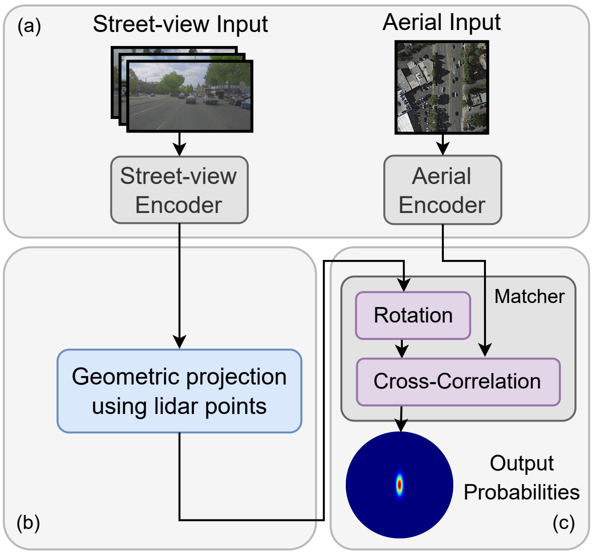


Fig. 5.7: Overview of the lidar-based pseudo-label model architecture. Same as Fig. 5.2 with (b) replaced by a geometric projection utilizing a lidar point cloud.

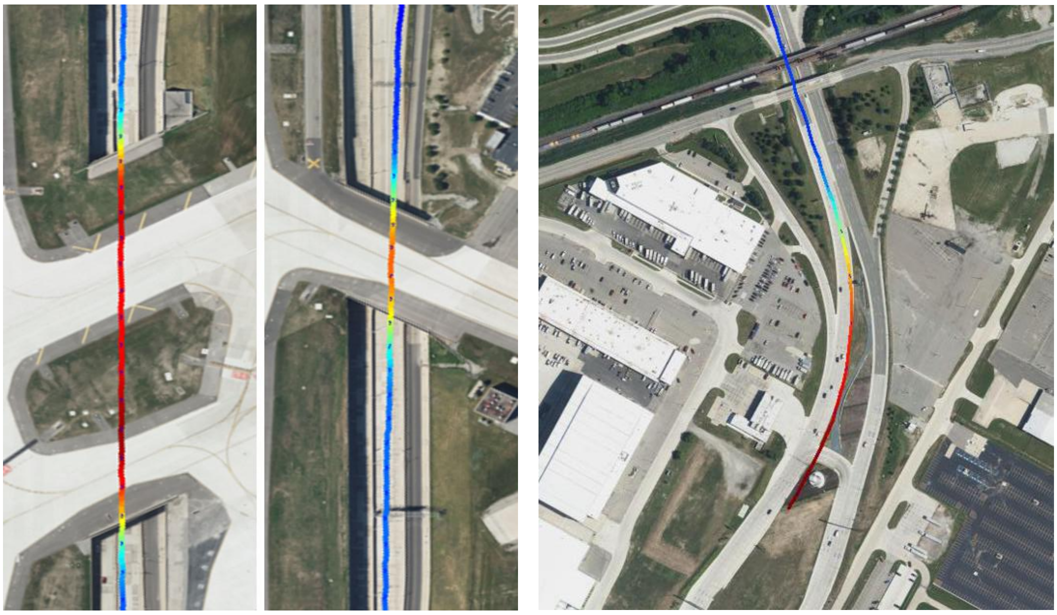
as well as the relative transformations between subsequent frames that are included in the datasets. The relative transformations are typically determined using a (visual or lidar-based) SLAM system by the authors of the datasets that yields accurate short-term trajectories, but is subject to drift up to the GNSS noise over the long-term. We thus utilize the relative transformations between subsequent frames with high confidence, and the pseudo-labeled geo-poses with low confidence. This causes the random, non-systematic error in the predicted poses to average out over sufficiently long sequences. The pose graph is optimized using a least-squares approach [47]. The resulting poses are used as the final pseudo-labels.

Step 5 We verify the accuracy of the resulting labels over a subset of the data using visualizations as in Fig. 5.6.

5.4.4 Data Pruning

The autonomous driving datasets contain samples that are not usable for cross-view matching tasks, for instance if the vehicle is traveling through a tunnel or if the street-view and aerial images are significantly out of date (*cf*. Fig. 5.8). We exclude these samples from the training using an automatic data-pruning approach as follows.

We identify invalid data by inspecting the covariance matrices that are predicted by the pseudo-labelling model for each sample. If the model is able to yield a low uncertainty of the vehicle pose (*i.e.* if it is able to gain some amount of information through the matching process), the data sample is assumed to be valid. If the uncertainty is high, the data sample is defined as invalid.



(a) Tunnel and overpass.

(b) Street-view trajectory and aerial image are out of date due to a reconstructed road.

Fig. 5.8: Examples of vehicle frames that are not usable for cross-view matching task. The line shows the driven trajectory. The line's color indicates the outlier score from low (blue) to high (red). Outliers are pruned from the training split. Vehicle data: Ford AV [3]. Map Data: Bing Maps © 2022 TomTom, © Vexcel Imaging.

We define the outlier score γ for each sample as the generalized variance [64] of the corresponding prediction P:

$$\gamma = \det(\mathrm{Var}(P)) \tag{5.17}$$

We further apply a one-dimensional Gaussian filter to the outlier scores along each trajectory, since invalid samples are typically found along contiguous sections of the trajectory. We sort all samples by the outlier score and prune the 1% of samples with the highest score.

5.4.5 Dataset Splits

For the main experiments, we use several training and test splits that represent different evaluation settings.

Same-area and same-vehicle The first setting follows the protocol introduced by Shi and Li [107] and uses a subset of the first two scenes in Ford AV data as test split. The training split consists of all scenes in Ford AV that are captured at a different time than the test split. Both splits cover the same area and are captured using the same type of vehicle and camera setup.

Cross-area and cross-vehicle For the second setting, we use the same test split in Ford AV as Shi and Li, but train on data from Argoverse V1 and V2, Lyft L5, Nuscenes and Pandaset. We remove samples from the training split that are captured in Detroit where Ford AV is recorded. The model therefore does not have access to data from the test region or test vehicle during training, and is required to generalize across these parameters.

Same-area and no-vehicle We propose a novel setting where the model is trained without access to the street-view data[5] and is therefore conditioned to learn a prior distribution of vehicle poses over the aerial image. We use the same training and test splits as Shi and Li.

Ablation studies For the ablation studies, we use scenes from all datasets that are captured in Palo Alto or San Francisco as the test split, and train on the remaining data, excluding KITTI-360 which does not contain full-surround view of cameras.

5.5 Evaluation

5.5.1 Implementation

The following sections provide details on the implementation of the method described in this chapter.

Main We train our main model with the *Base* variant of ConvNeXt [80] which is pretrained on ImageNet [29]. The street-view features F_{Si} are extracted with $c_S = 128$ channels. We use 4 heads in $n_{\text{blocks}} = 3$ self-attention and cross-attention blocks to project features from PV to BEV. Point pillars in the projection step are sampled with $z = 16$ points from $h_{\text{min}} = -5.0\text{m}$ to $h_{\text{max}} = 10.0\text{m}$. The BEV map is constructed with $c_B = 128$ channels and a resolution of 40×40, and is upsampled by a factor 8 to yield the final features F_B with $h_b = 40 \cdot 8 = 320$. The aerial image is sampled with a resolution of $h_A = 512$ and $0.3\frac{\text{m}}{\text{px}}$. We map both aerial and BEV feature maps to $c_A = 8$ channels before computing their matching score in Eq. (5.3).

The model is trained for 100k iterations with a batch size of 1. We employ the RectifiedAdam optimizer [78] with a learning rate of $1.0 \cdot 10^{-4}$ and polynomial decay schedule. The loss function is parametrized with standard deviations of $\sigma_t = 0.5\text{m}$ and $\sigma_\theta = 2°$. We apply decoupled weight decay with a magnitude of $1.0 \cdot 10^{-4}$ [81].

The prior pose T_A is chosen in a region of $40\text{m} \times 40\text{m}$ around the vehicle pose T_S with an orientation noise of up to $20°$, following Shi and Li [107].

Ablations studies We choose an experimental setup for the ablation studies with reduced computational cost to allow running more evaluations. We use the *Nano*

[5] We set all RGB values in the images to 0.

Tab. 5.2: Recall in percent on a subset of the first two scenes of the Ford AV dataset [3] following the evaluation protocol introduced by Shi and Li [107]. Results of previous works on Ford AV are provided by Shi and Li. The first three rows represent retrieval methods that are adapted for pose estimation. The initial pose is chosen in 40m × 40m around the vehicle with up to $20°$ of rotation noise. All methods are vision-based only. *Ours w/o SV* refers to the aerial prior model that is trained and tested without access to the street-view (SV) images.

				Scene 1						Scene 2					
	Cross-	Cross-	Multi-		Lateral			Longitudinal			Lateral			Longitudinal	
	area	vehicle	cam	1.0m	3.0m	5.0m	1.0m	3.0m	5.0m	1.0m	3.0m	5.0m	1.0m	3.0m	5.0m
CVM-Net	✗	✗	✗	9.1	25.7	41.3	4.8	13.2	21.9	9.8	28.6	47.1	4.2	11.8	20.3
SAFA	✗	✗	✗	9.3	28.7	48.0	4.3	11.8	20.1	11.2	34.1	53.4	5.0	13.4	22.9
DSM	✗	✗	✗	12.0	35.3	53.7	4.3	12.5	21.4	8.5	24.9	37.6	3.9	12.2	21.4
VIGOR	✗	✗	✗	20.3	52.5	70.4	6.2	16.1	25.8	20.9	54.9	75.7	6.0	16.9	27.0
Shi and Li	✗	✗	✗	46.1	70.4	72.9	5.3	16.4	26.9	31.2	66.5	78.8	4.8	15.3	25.8
Ours w/o SV	✗	-	-	15.1	51.3	72.0	5.0	15.2	24.4	11.3	37.8	62.2	4.7	15.3	26.0
Ours	✗	✗	✗	**87.8**	**98.4**	**99.6**	**67.7**	**93.5**	**94.0**	**73.5**	**94.2**	**96.1**	**42.2**	**86.0**	**87.9**
Ours	✗	✗	✓	96.3	99.6	99.6	76.0	95.3	96.0	88.0	99.9	100.0	58.9	93.3	93.6
Ours	✓	✓	✗	60.9	86.5	93.3	19.2	52.1	56.8	49.5	83.0	88.7	19.3	44.7	48.6
Ours	✓	✓	✓	77.0	96.2	97.6	24.0	67.6	76.1	73.0	96.5	97.8	25.6	61.7	69.4

variant of ConvNeXt [142] as encoder in the model architecture. Aerial images are sampled with a smaller resolution of $h_A = 256$ at $0.5\frac{\text{m}}{\text{px}}$, and the BEV map is constructed with $c_B = 32$ channels at resolution 48×48 and upsampled by factor 4.

The prior pose T_A is chosen up to 30m away from the the vehicle pose T_S with an orientation noise of up to $10°$. All other parameters are kept as in the main evaluation if not stated otherwise.

5.5.2 Metric

We evaluate our method using the longitudinal and lateral recall with a distance threshold of r meters. A prediction along each dimension is deemed correct if the respective location error is less than r. We report lateral and longitudinal recall for $r \in \{1.0\text{m}, 3.0\text{m}, 5.0\text{m}\}$ following previous works [107].

5.5.3 Results

We report the main results of our work and baselines on the Ford AV test split in Tab. 5.2 and make the following observations:

- Our model vastly outperforms previous works in the evaluation protocol of Shi and Li [107], *i.e.* same-area, same-vehicle and single-cam. For instance, existing methods achieve the best longitudinal recall with $r = 1.0$m of 6.0% and 6.2% on the two scenes, which increases to 67.7% and 42.2% in our work. Overall, our method correctly localizes more than 90% of frames to within 3.0m of the ground-truth location in this setting.
- The model achieves a lower localization accuracy in the challenging cross-area and cross-vehicle setting than in the same-area and same-vehicle setting, since

Tab. 5.3: Recall in percent on all six trajectories of the Ford AV dataset [3] (2017-08-04 V2) w.r.t. our pseudo-labeled ground-truth in a cross-area and cross-vehicle setting. The prior pose is chosen with up to 50m error to the vehicle position. Orientation noise is defined below.

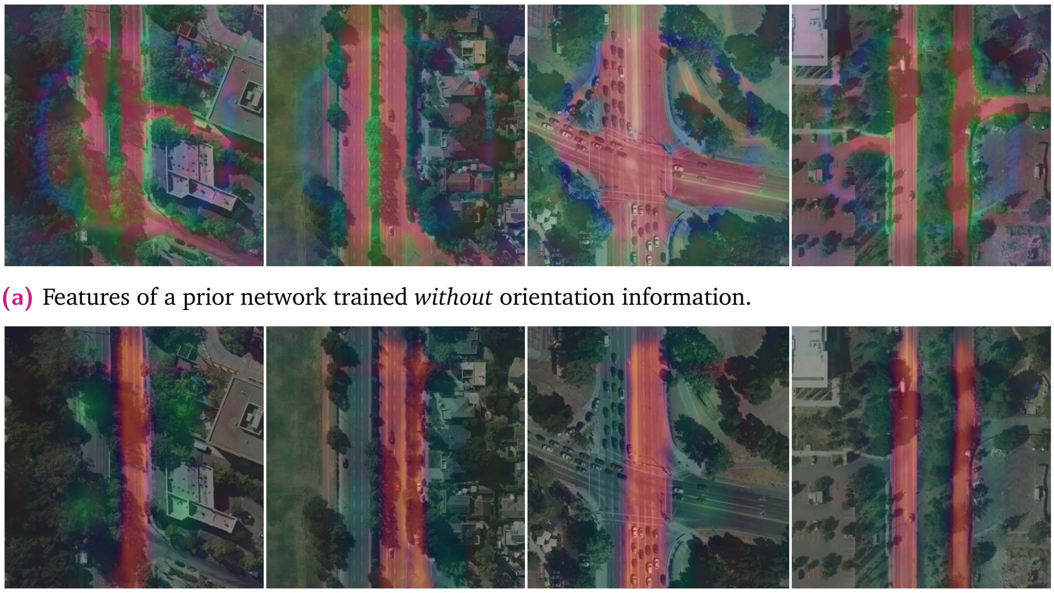
	Angle	Lateral			Longitudinal		
	noise	1.0m	3.0m	5.0m	1.0m	3.0m	5.0m
Scene 1	30°	63.8	87.4	91.1	27.4	61.1	67.6
Scene 2	30°	58.7	83.2	85.6	21.8	53.4	60.7
Scene 3	30°	90.1	99.2	99.5	77.5	98.1	99.0
Scene 4	30°	89.9	99.7	100.0	71.6	95.8	97.8
Scene 5	30°	89.5	99.8	99.8	78.7	98.5	98.9
Scene 6	30°	87.8	97.9	98.3	69.9	94.1	95.3
Mean	30°	80.0	93.5	95.7	57.8	83.5	86.0
Scene 1	360°	54.8	72.7	75.9	23.2	53.1	59.7
Scene 2	360°	44.1	62.8	65.4	17.4	41.3	48.0
Scene 3	360°	90.3	98.8	99.0	77.3	97.4	98.3
Scene 4	360°	89.3	98.9	99.4	70.8	94.6	97.2
Scene 5	360°	89.5	99.7	99.8	79.2	98.2	98.5
Scene 6	360°	86.6	96.1	96.7	70.2	93.5	94.8
Mean	360°	75.8	88.2	89.4	56.4	79.7	82.3

it has to generalize to an unseen target region and camera setup. Remarkably, the performance is still higher than existing works that are trained on same-area and same-vehicle data.

- The recall is improved significantly when using all surround cameras, rather than only the front camera. More than 95% of all frames are localized to within 3.0m when using multiple cameras in the same-area and same-vehicle setting. Fig. 1.5 shows example predictions of our model using all available cameras in a cross-area and cross-vehicle setting.

- The longitudinal recall tends to be lower than the lateral recall in all settings. Vehicles are typically aligned with roads that have a repeating structure in the longitudinal direction which increases the uncertainty of the prediction along this axis. In contrast, the lateral direction is often characterized by discriminative features such as the curb, side-walk or buildings that allow for more accurate localization along this axis.

The experiments shown in Tab. 5.2 follow an existing evaluation protocol by Shi and Li [107] and allow comparing our method with previous works. However, their test split contains only a subset of the first two scenes in Ford AV [3] which were identified by the authors to have a low ground-truth error. We therefore additionally evaluate our method on all six scenes in Ford AV w.r.t. our pseudo-labeled ground-truth and report results in Tab. 5.3. Overall, the model achieves a median location error of 0.87m both with and without known orientation.



(a) Features of a prior network trained *without* orientation information.

(b) Features of a prior network trained *with* orientation information. Driving direction points upwards.

Fig. 5.9: Visualization of features learned by the prior network that is trained without access to street-view data. The images show the feature map predicted for the aerial image, reduced to three channels via a Principal Component Analysis (PCA) and mapped onto RGB. Vehicle data: Lyft L5 [56]. Map Data: Bing Maps © 2022 TomTom, © Vexcel Imaging.

5.5.4 Aerial Prior

We propose a novel setting where the model is trained without access to the street-view data and therefore learns a prior distribution of vehicle poses on the aerial image. The results are shown in Tab. 5.2.

We find that three existing methods, CVM-Net [57], SAFA [108] and DSM [110], that are adapted from the retrieval task achieve a localization accuracy that is *worse* than the prior network. Similarly, the two best existing pose estimation methods of Zhu *et al*. [163] and Shi and Li [107] achieve a longitudinal recall that is nearly identical with the prior network, and yield better localization only in the lateral direction. This suggests that pose estimation networks in existing works might rely mainly on prior information that is available on the aerial image, rather than on cross-view matching.

To further investigate the prior network, we visualize the features it predicts for aerial images in Fig. 5.9. We find that the features tend to depict a segmentation of the image into drivable and non-drivable regions. Drivable regions are shown in red, which corresponds the first principal component of the feature distribution.

We additionally find that the network appears to have gained knowledge about the right-hand traffic that the training regions of Germany and the US follow. When training with a known orientation of the vehicle (*i.e.* pointing upwards in the image)

Tab. 5.4: Ablation studies tested on all scenes from Palo Alto and San Francisco with our pseudo-labeled ground-truth. The initial pose is chosen randomly up to 30m from the vehicle with up to 10° of rotation noise. ME: Mean error in meters. RMSE: Root mean squared error in meters.

Method modification	ME	RMSE
–	1.19	3.44
No deformable offsets	1.22	3.65
No ResNet blocks at stride 1	1.22	3.51
$n_{\text{heads}} = 1$	1.23	3.59
No deformable offset skip connection	1.23	3.69
No data pruning	1.23	3.62
No MLPs in transformer blocks	1.23	3.52
$n_{\text{blocks}} = 1$	1.24	3.76
No attention skip connection	1.24	3.77
AV $\rightarrow$ QKV attention	1.25	3.63
No Self Attention Block	1.38	4.21
No pseudo-labels	2.37	5.15
No BEV upsampling	2.63	5.87
No encoder pretraining	4.15	9.21
No street-view data	11.95	15.20

the first principal component of the features now highlights only the right-hand lanes of a road (*cf*. first three images in Fig. 5.9b). This characteristic does not arise when training with 360° of orientation noise (*cf*. Fig. 5.9a). If the lanes are too far apart where they cannot distinctly be identified as opposing lanes of the same road, the features also highlight both lanes (*cf*. last image in Fig. 5.9b).

Our proposed model is the first learned method for cross-view pose estimation with a localization accuracy that is significantly higher than the prior localization both in the lateral and longitudinal dimensions.

5.5.5 Ablation Studies

In this section, we evaluate the contribution of various components of our method to the overall localization accuracy by conducting a series of ablation studies. Each study involves removing or modifying a specific component of the method and assessing the resulting impact on the performance. The results are summarized in Tab. 5.4.

All investigated components improve the resulting localization accuracy of our method. The most important factors are pretraining the encoder ($\Delta\text{ME} = 2.96$), upsampling the BEV map before matching with the aerial image ($\Delta\text{ME} = 1.44$), and including the self-attention block when constructing the BEV ($\Delta\text{ME} = 0.19$).

Training and testing the model with the original, noisy ground-truth (*i.e. No pseudo-labels* in Tab. 5.4) results in a larger error than training and testing with our pseudo-

Tab. 5.5: Mean Absolute Trajectory Error (ATE) in meters of the trajectory estimation on scenes in the KITTI-360 dataset [74]. We compare our drift-free method with the best performing visual and lidar odometry methods on the KITTI-360 leaderboard. ⋆: Since CT-ICP [28] and SOFT2 [27] are subject to drift and report the error as Relative Trajectory Error (RTE) in percent, we approximate their ATE by multiplying the RTE with the sequence length in meters.

Method	00	02	03	04	05	06	07	09	10	Mean
Ours	0.62	0.80	1.01	0.71	0.62	0.80	0.60	0.67	2.12	0.78
CT-ICP* [28] (Lidar)	34.41	58.24	4.69	64.76	18.26	33.48	16.62	47.56	23.07	43.93
SOFT2* [27] (Visual)	24.67	55.94	3.13	45.13	14.23	28.22	14.96	52.00	29.12	39.10

labeled ground-truth. Since the model is trained in a cross-area setting and does not see ground-truth from the test region during training, the improvement gained from utilizing our pseudo-labels cannot be due to the model overfitting on potentially incorrect pseudo-labels in the test region.

5.5.6 Trajectory Estimation

We evaluate our proposed tracking framework on scenes from the KITTI-360 dataset [74] in a cross-area and cross-vehicle setting. The dataset contains one forward and two sideways facing cameras. We follow the evaluation protocol of VO methods on KITTI-360: The predicted trajectory is aligned with the ground-truth trajectory via a rigid transformation, and the error is measured w.r.t. the aligned trajectory. Following this protocol allows evaluating our method without relying on the original, noisy geo-poses, or on our pseudo-labeled geo-poses.

Tab. 5.5 shows the Absolute Trajectory Error (ATE) of our method on scenes in KITTI-360. The method achieves a mean error of 0.78m over all samples in all trajectories. When measuring the localization w.r.t. our pseudo-labeled ground-truth, the ATE is nearly identical at 0.85m, which supports the quality of the pseudo-labels.

State-of-the-art odometry methods such as CT-ICP [28] and SOFT2 [27] achieve a mean Relative Trajectory Error (RTE) of roughly 0.4% on KITTI-360, *i.e.* around 0.4m after traveling for 100m. However, this drift accumulates over longer sequences and results in much larger ATE due to the average sequence length of 7.4km. This highlights the benefit of using our cross-view matching to avoid drift on long-term trajectories. When employing state-of-the-art odometry methods with a mean RTE of 0.4%, the accumulated drift exceeds the ATE of our method after traveling for just 200m on average.

We provide videos that demonstrate the application of our method on two scenes in Ford AV and KITTI-360 at the following url:

`https://fferflo.github.io/projects/vismetcvgl23`

5.6 Summary

In this chapter, we present the first end-to-end trainable model for cross-view pose estimation that requires only camera images as input and achieves sub-meter accurate predictions. The model utilizes a BEV map that represents the vehicle environment and is matched with an aerial image to predict a probability distribution over possible poses. We further propose a novel filtering framework that allows integrating the model's predictions over time to estimate the long-term trajectory of a platform.

We use datasets from the autonomous driving field and orthophotos to train and test the model, and apply pseudo-label and data pruning steps to preprocess the data and obtain accurate ground-truth geo-poses. Experimental evaluation shows that the method achieves a per-frame median position error on the Ford AV dataset of less than a meter, and a mean position error over trajectories in KITTI-360 of less than a meter. Remarkably, while existing methods train in a *same-area* setting, our model achieves better localization even when trained in a *cross-area* setting, *i.e.* without access to training data from the test region.

Our contributions are summarized as follows:

Model We design a novel end-to-end trainable model that requires only camera images as input and predicts a probability distribution over possible platform poses on an aerial image. Unlike in previous works, the model does not require three-dimensional point clouds (*e.g.* captured from lidar or radar) as input. It predicts a soft, uncertainty-aware distribution over poses, rather than a single hard pose.

Dataset We combine multiple datasets from the autonomous driving sector with aerial images from multiple orthophoto providers to train and evaluate our method. We apply pseudo-label and data pruning approaches to clean the dataset and provide accurate ground-truth labels. The dataset covers multiple cities in Germany and the US and is captured with a range of vehicles and camera setups. This allows among others evaluating in both same-area and cross-area settings on the same test split.

Aerial prior We propose a novel evaluation setting in which the model is trained and tested without access to street-view data and thus estimates a prior distribution of vehicle poses over the aerial image. This allows assessing how much of a method's performance is due to (1) prior information on the aerial image and (2) effective cross-view matching between both views. Our model is the first to achieve significantly better localization than the aerial prior in both the longitudinal and lateral directions.

Trajectory estimation We propose a novel method to integrate the uncertainty-aware predictions of the model in an EKF to estimate the trajectory of a platform over time. We use the prior pose estimate of the EKF to act as a soft windowing function in the model's multi-modal prediction, and compensate for the reduced covariance of the soft windowing to avoid self-reinforcing confidence. Our contribution allows employing a Kalman filter with the multi-modal model predictions and avoids the need for a computationally expensive particle filter.

6 Presenting a Unified View on Retrieval and Pose Estimation

6.1 Motivation

In the previous chapters, we presented a decomposition of CVGL into two independent sub-problems, *i.e.* retrieval and pose estimation. This division dominates the research field: Most works address either the retrieval, or the pose estimation task, but not both. The existing problem formulations and model architectures used for the two types of problems also diverge significantly.

In the following paragraphs, we briefly analyze our formulations of retrieval and pose estimation presented in Chapter 4 and Chapter 5, and identify their common ground to motivate a new approach that addresses the two problems jointly. Both types of methods follow the same following structure for localization:

1. *Partition the search region into a discrete set of hypotheses.*

   Let $\mathcal{R}_0 \subset \mathrm{SE}(2)$ be the entire search space, *i.e.* the continuous set of potential camera poses that are under consideration. Each hypothesis h corresponds to a continuous subset $\mathcal{R}(h) \subset \mathcal{R}_0$ of the search space. The set of all hypotheses $\mathcal{H}$ provides a dense, non-overlapping coverage of the search space:

   $$\mathcal{R}_0 = \bigsqcup_{h \in \mathcal{H}} \mathcal{R}(h) \tag{6.1}$$

   In retrieval, the set $\mathcal{R}(h)$ for a hypothesis h corresponds to locations in a single $l \times l$ cell with any possible camera orientation $\theta \in [0, 2\pi)$.

   In pose estimation, the hypotheses are chosen in a regular grid over the search space with a step size equal to the aerial image's pixel resolution, and include orientations $\theta \in \Theta$ as a third DoF (*cf*. Eq. (5.5)). The set $\mathcal{R}(h)$ thus corresponds to a small range of camera locations and orientations.

   Given a Probability Density Function (PDF) $p : \mathcal{R}_0 \rightarrow \mathbb{R}$ over possible camera poses in the search region, the corresponding probability of a discrete hypothesis h in retrieval or pose estimation is defined as follows:

   $$P(h) = \int_{\mathcal{R}(h)} p(x)dx \tag{6.2}$$

2. *Assign an embedding $f_A^{(h)} \in \mathbb{R}^c$ to each hypothesis h. Compute the predicted score for a hypothesis as the dot-product between its embedding and the street-view embedding $f_S \in \mathbb{R}^c$.*

   In retrieval, the aerial images for a given cell h and street-view photo are directly mapped onto the embedding vectors $f_A^{(h)}$ and f_S (*cf*. Sec. 4.3.3). The score assigned to a given hypothesis is computed as the dot-product between the corresponding embeddings in Eq. (4.5).

   In pose estimation, the model instead predicts structured features maps F_A for the aerial image and F_B for a BEV representation of the street-view image(s). The features $F_A^{(h)}$ for a hypothesis h are determined by rotating and translating F_A according to the mean pose in $\mathcal{R}(h)$ (*cf*. Sec. 5.3.2). This allows computing the score for a hypothesis h as the dot-product between $F_A^{(h)}$ and F_B.

   While the embeddings $F_A^{(h)}$ and F_B are defined as three-dimensional feature maps with a given height, width, and number of channels, the dot-product is invariant to the dimensionality. The score for h is determined equally (1) by computing the three-dimensional dot-product between F_B and $F_A^{(h)}$, or (2) by flattening the feature maps into vectors $f_A^{(h)}$ and f_S in $\mathbb{R}^c$ and computing their one-dimensional dot-product, similar to the cross-view retrieval.

3. *Use an optimized algorithm to evaluate multiple hypotheses jointly.*

   In retrieval, a reference database of all hypotheses is built that allows searching through the embeddings using approximate nearest neighbor algorithms. Multiple hypotheses are tested jointly by querying the database once with the given street-view embedding.

   In theory, this approach would also be applicable to the pose estimation method: The feature maps $F_A^{(h)}$ for all hypotheses h are flattened and inserted into a retrieval-like reference database which is then queried using the flattened F_B. In practice, however, a cross-correlation in Fourier domain between $F_A^{(\theta)}$ and F_B for the set of orientations $\theta \in \Theta$ allows for much faster, real-time performance and avoids the construction of a dedicated reference database.

Both retrieval and pose estimation follow the above structure to localize street-view images against a reference database of aerial imagery. The main difference in the formulation lies in the size of the search space $\mathcal{R}(h)$ covered by a single hypothesis h.

In pose estimation, $\mathcal{R}(h)$ covers only a small range of camera locations corresponding to the pixel resolution of the aerial image (*e.g.* $30\frac{\text{cm}}{\text{px}}$ in Sec. 5.5.1), and a small range of orientations corresponding to the discretization of the angle dimension into Θ.

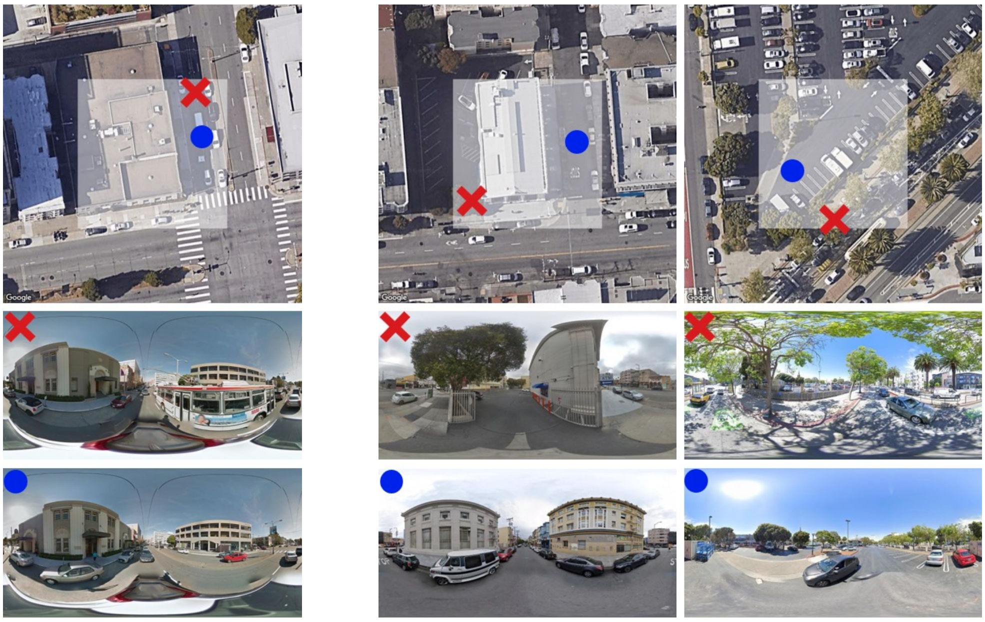

(a) Similar street-view images. (b) Dissimilar street-view images.

Fig. 6.1: Examples of aerial images with paired street-view images in the VIGOR dataset [163]. *Top row:* Aerial image with the location of two street-view cameras. *Bottom rows:* Street-view photos taken from the marked locations. Search regions are shown in white. Photos taken from different poses on the aerial image potentially lead to vastly differing street-view appearance, but have to be mapped to embeddings that match with the same aerial image.

Photos captured from different possible poses in $\mathcal{R}(h)$ thus likely have very similar visual appearance.

In contrast, a retrieval hypothesis h covers a much larger region of camera poses $\mathcal{R}(h)$, *i.e.* a cell of size $l \times l$ with $l = 30$m. Street-view photos that are captured from different poses in the cell potentially have a vastly differing appearance (*cf*. Fig. 6.1). The retrieval model however is required to map the photos from all poses to embeddings that align with the single embedding of the cell, which represents a significant challenge than in cross-view pose estimation.

To address this issue, we consider a solution to the retrieval problem that decomposes a retrieval hypothesis into multiple pose estimation hypotheses. These hypotheses are easier to test, since they reduce the requirement for the model to predict matching embeddings over vastly different camera poses, and benefit from the strong geometric bias of PV2BEV and BEV-based matching. The individual pose estimation hypotheses are then fused in a probabilistic manner to arrive at the score of the original retrieval hypothesis.

6.2 Overview

Following the above motivation, we propose a novel approach for CVGL in this chapter, *i.e.* Contrastive Bird's Eye View Training (C-BEV), that addresses the retrieval problem by representing it as an unsupervised pose estimation task. The method consists of an end-to-end differentiable model that outputs a retrieval score for a pair of street-view and aerial images, but internally utilizes the BEV-based matching introduced in Chapter 5. The model is trained in a regular retrieval setting with a pairwise assignment of street-view and aerial images as ground-truth. Remarkably, it learns to predict accurate camera poses, *despite being supervised only with the retrieval loss*. We integrate the method in a two-stage localization pipeline that follows the retrieve-and-rerank formulation similar to strategies employed in SVGL [51, 134, 161], and significantly improves the recall of baseline methods.

In Sec. 6.3, we describe the BEV-based retrieval and how it is used in a two-stage retrieve-and-rerank pipeline. In Sec. 6.5 and Sec. 6.6, we give an overview on several benchmark datasets and provide experimental results both for the improved retrieval, as well as the unsupervised pose estimation.

This chapter is based on our publication in CoRR 2023 [35].

6.3 Method

6.3.1 Overview

We propose to decompose the retrieval problem into multiple pose estimation subproblems. Given a retrieval hypothesis $h \in \mathcal{H}_{\text{retrieval}}$, *i.e.* a cell in a large RoI, we choose a set of camera pose hypotheses $f(h) \subset \mathcal{H}_{\text{pose-estimation}}$ in a dense grid over the cell and compute the probability of h as the sum of probabilities of individual poses:

$$P(h) = P(\bigvee_{h' \in f(h)} h') \overset{1}{=} \sum_{h' \in f(h)} P(h') \tag{6.3}$$

A retrieval hypothesis h is thus evaluated by explicitly testing potential camera poses within the cell using our pose estimation approach. The expression $f(h)$ defines the set of camera poses that are tested and must provide a sufficiently dense coverage of the cell. It is defined as

$$f(h) = \mathcal{R}^{-1}_{\text{pose-estimation}}(\mathcal{R}_{\text{retrieval}}(h)) \tag{6.4}$$

[1] $P(h'_1 \wedge h'_2) = 0$ for any pose hypotheses $h'_1 \neq h'_2$.

where $\mathcal{R}_{\text{retrieval}}$ maps a retrieval hypothesis h onto the respective subset of SE(2), and $\mathcal{R}^{-1}_{\text{pose-estimation}}$ decomposes that subset into pose estimation hypotheses as described in Sec. 5.3.2.

While our method explicitly computes probabilities $P(h')$ for different camera poses $h' \in f(h)$, we supervise only the final retrieval score $P(h)$ in our loss function. The probabilities $P(h')$ represent intermediate activations in the network that are trained end-to-end via the retrieval loss, *but are not supervised directly*. Remarkably, the model nevertheless learns to predict accurate poses despite never seeing pose ground-truth during training.

6.3.2 Two-stage Pipeline

The decomposition of retrieval hypotheses into pose estimation hypotheses significantly increases the total number of hypotheses that have to be considered for a given RoI. While ordinary retrieval uses only a single hypothesis and embedding per cell, choosing the grid of poses over the cell *e.g.* with 64^2 translations and 32 orientations already leads to a factor of roughly 10^5 more hypotheses. Instead of inserting the embeddings for all pose hypotheses into a retrieval-like reference database that is queried with a nearest neighbor approach, we employ a retrieve-and-rerank paradigm as follows. The localization of a query photo is split into two subsequent stages:

Stage 1: Retrieve Use a classical retrieval method to find the top-k cells for the given query in the reference database. The retrieval is able to efficiently search through vast amounts of data, but potentially has a low recall. We consider the k highest-ranked cells as candidates that are forwarded to the second stage.

Stage 2: Rerank Recompute the retrieval score for each candidate following Eq. (6.3) and rerank the list of candidates according to the new scores. This improves the recall at the cost of more expensive matching. Additionally, evaluating Eq. (6.3) implicitly performs pose estimation of the query photo on the matching aerial images by computing $P(h')$ for all pose hypotheses h'.

The cost of retrieval and reranking grows sub-linearly and linearly w.r.t. to the number of cells, respectively. The number of candidates represents a trade-off between the relative improvement in recall, and additional matching cost.

The retrieve-and-rerank paradigm has been used extensively in the field of SVGL [51, 134, 161] where the retrieval stage is often followed by a geometric verification of the candidates. Since camera poses in SVGL are typically estimated with six DoF, densely sampling hypotheses over the pose space in this case is not computationally feasible. Instead, they typically rely on a stochastic method to verify matches, *i.e.* Random Sample Consensus (RANSAC) [39]. Our second stage similarly performs geometric verification by explicitly considering the camera pose, but is able to do so densely due to the smaller search space with three DoF.

We utilize the existing state-of-the-art method Sample4Geo [30] as the first stage, and a BEV-based matching model for the second stage that is based on our work in Chapter 5. The following sections provide an overview of the two stages.

6.3.3 Stage 1 - Retrieve

For the first stage, we train a retrieval model in a setting following Sample4Geo [30]. This allows for a fair evaluation of our proposed second stage w.r.t. the recent state-of-the-art. Since their work shares a common structure with our retrieval method (*cf*. Chapter 4), we describe the main differences below.

Sample4Geo uses the ConvNeXt backbone [80] for extracting features from the input images. Unlike in our work, it uses a simple mean-pooling layer to fuse local features into a single embedding representation, and shared weights between the aerial and street-view domain. The model is trained on existing benchmark datasets (*cf*. Tab. 4.3) and follows their choice of aerial images and search region layout; *i.e*. an aerial image at a single LoD, and the one-to-one and many-to-many matching formulations.

We use the HEM strategy of Sample4Geo which recomputes the embeddings of all samples in the dataset every few epochs and keeps the result in memory to mine hard reference images. During the first few epochs, the haversine distance between the images' geolocations is used as distance metric.

6.3.4 Stage 2 - Rerank

In the second stage, we decompose the retrieval hypothesis into individual camera pose hypotheses according to Eq. (6.3). The camera poses are tested following our pose estimation approach presented in Chapter 5. The scores of all poses are then fused to yield the overall retrieval score. The individual pose probabilities represent intermediate activations in the network that are trained end-to-end via the retrieval loss, but are not supervised directly.

We evaluate our approach on benchmark datasets that contain panorama images rather than pinhole images as queries, and do not provide ground-truth poses over six DoF that would allow constructing point pillars as described in Sec. 5.3.2. We therefore choose a simpler method for the Perspective View to Bird's Eye View (PV2BEV) step of the model as follows.

Perspective View to Bird's Eye View Given the feature map $F_S \in \mathbb{R}^{h_S \times w_S \times c_S}$ with height h_S, width w_S and c_S channels that is extracted for the street-view panorama, we project it into a BEV representation as follows (*cf*. Fig. 6.2).

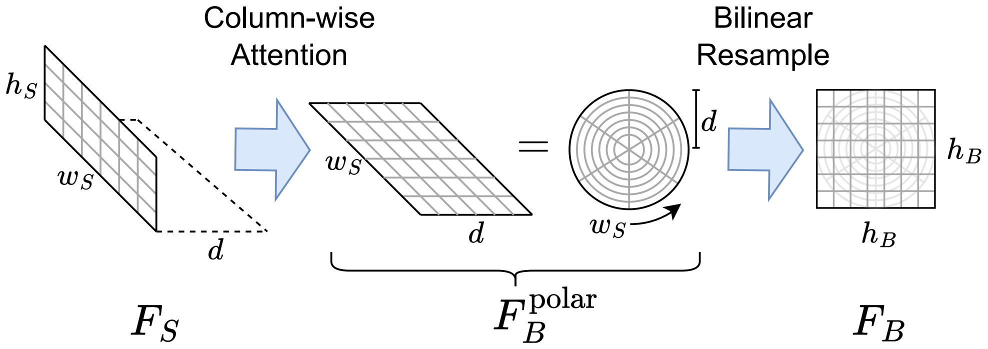


Fig. 6.2: Our method for transforming the feature map $F_S \in \mathbb{R}^{h_S \times w_S \times c_S}$ of a street-view panorama into a BEV representation $F_B \in \mathbb{R}^{h_B \times h_B \times c_B}$. F_S is first transformed into a polar BEV map $F_B^{\text{polar}} \in \mathbb{R}^{w_S \times d \times c_S}$ via column-wise attention over d discrete depth values. $F_B^{\text{polar}} \in \mathbb{R}^{w_S \times d \times c_S}$ is then resampled bilinearly into the cartesian representation F_B.

The features F_S are first transformed into a polar BEV map $F_B^{\text{polar}} \in \mathbb{R}^{w_S \times d \times c_S}$ via a column-wise attention operation. Each cell in the polar representation corresponds to a particular distance and horizontal viewing angle w.r.t. the camera pose. We consider w_S angles based on the horizontal resolution of the panorama, and d evenly spaced depth values between d_{min} and d_{max}.

For each angle, we define the d cells in F_B^{polar} as queries, and the h_S features along the image column in F_S as keys and values of an attention operation. This allows each query cell to gather information from the corresponding column in the panorama's feature map. The d query features are defined as learnable parameters and are shared across all image columns.

Since F_B^{polar} is given in a polar representation (*i.e.* with cells being indexed by angle and distance w.r.t. camera pose), we transform it into cartesian representation (*i.e.* with cells being indexed by lateral and longitudinal distance w.r.t. the camera pose) via bilinear interpolation. The resulting features F_B are then compared with the aerial features to test different camera pose hypotheses using a cross-correlation operation as in Sec. 5.3.2. We normalize both F_B and F_A globally via their L2-norm before applying the matching operation.

Retrieval score The matching scores between the BEV features F_B and aerial features F_A represent the logits $\ell(h')$ (*i.e.* unnormalized log-probabilities) of camera pose hypotheses h'. The logit $\ell(h)$ of the overlying retrieval hypothesis h is computed as the fusion of all $h' \in f(h)$ by applying (6.3) as follows:

$$\ell(h) \overset{2}{\equiv} \log P(h) \overset{(6.3)}{=} \log \sum_{h' \in f(h)} P(h') \equiv \log \sum_{h' \in f(h)} \exp \ell(h') = \underset{h' \in f(h)}{\text{LSE}}\, \ell(h') \qquad (6.5)$$

[2] We write $\equiv$ to indicate equality up to an additive constant. An unnormalized log-probability ℓ and the corresponding log-probability $\log P$ are equal up to an additive constant. The constant is removed by normalizing logits via a softmax operation.

Eq. (6.5) computes the same decomposition of h into $f(h)$ as Eq. (6.3), but operates on logits ℓ rather than probabilities P. The log-sum-exp (LSE) operation in Eq. (6.5) has the same purpose as $\sum$ in Eq. (6.3). Explicitly evaluating the *exp* and *log* terms however potentially results in numerical instability during training. To address this problem, deep learning frameworks such as Jax [16] and PyTorch [98] provide a dedicated method that computes LSE in a numerically stable way.

Score fusion Eq. (6.5) computes the score $\ell_{\text{stage2}}(h)$ assigned to a given hypothesis using only information gained from the second stage's model. However, the first stage's prediction $\ell_{\text{stage1}}(h)$ already contains useful prior information for all candidate hypotheses that would be discarded if the reranking is performed solely w.r.t. $\ell_{\text{stage2}}(h)$.

We propose integrating the scores from both stages via a probabilistic fusion as follows. We employ P_{stage1} as prior probabilities that are updated via the probabilities P_{stage2} using Bayes' theorem, resulting in the fused probabilities P_{final}:

$$P_{\text{final}}(h) \propto P_{\text{stage2}}(h) \cdot P_{\text{stage1}}(h) \propto \exp(\ell_{\text{stage2}}(h) + \ell_{\text{stage1}}(h)) \tag{6.6}$$

The second stage is thus defined to finetune the predictions from the first stage, rather than rerank candidates from scratch.

Importantly, the fusion is performed both during training and testing of the second stage's model. It is therefore conditioned to specialize on the failure cases of the first stage in each batch, *i.e.* where the scores P_{stage1} represent an incorrect prediction. In contrast, any positive or negative query-reference pair in a batch that is correctly assigned a high or low $P_{\text{stage1}}(h)$, respectively, results in a low contribution to the overall loss, and thereby low supervision for $P_{\text{stage2}}(h)$.

6.4 Training Setup

We use similar settings to train the first and second stages. We use batches of matching pairs of street-view images and search cells analogous to CLIP [101], Sample4Geo [30], and our Chapter 4. We compute the pairwise similarity between all samples via a simple dot product in the first stage, and via decomposition into the BEV-based matching in the second stage (*cf*. Eq. (6.5)). We supervise the models using the symmetric cross-entropy loss with the same temperature and label smoothing.

Mining hard examples during training using the BEV-based scores is not computationally feasible, due to the quadratic cost of comparing samples in a large dataset. Instead, we utilize the embeddings predicted from the first stage to mine hard samples in the second stage. This allows for an efficient HEM, and provides additional

Tab. 6.1: Side-length (in meters) of aerial images in datasets used in this chapter. Each row corresponds to a different train-test split. All images per dataset have the same size in pixels.

	Training images		Testing images	
	Min.	Max.	Min.	Max.
VIGOR same-area [163]	64.3	75.3	64.3	75.3
VIGOR cross-area [163]	64.3	72.3	71.0	75.3
CVUSA [144]	95.1	203.7	95.2	203.5
CVACT-Val [77]	72.8	72.8	72.8	72.9
CVACT-Test [77]	72.8	72.8	72.8	72.9
CVUSA → CVACT	95.1	203.7	72.8	72.9
CVACT → CVUSA	72.8	72.8	95.2	203.5

supervision for the model to specialize on the failure cases of the first stage over the entire mining pool.

6.5 Data

To evaluate our proposed two-stage pipeline and compare with state-of-the-art methods, we conduct experiments on the benchmark datasets VIGOR [163], CVUSA [144] and CVACT [77]. Following Deuser *et al.*, we resize aerial images to a resolution of 384×384, and street-view panoramas to 384×768 in VIGOR and 140×768 in CVUSA and CVACT. We additionally evaluate in the cross-dataset settings CVUSA→CVACT and CVACT→CVUSA as proposed by Yang *et al.* [152], where the model is trained on a dataset and tested on the other dataset's validation split.

Pixel resolution As described in Sec. 2.4, existing datasets in the field of CVGL sample aerial images from a Mercator projection of the earth's surface, and are therefore subject to a scale inconsistency across different geographical regions. For instance, the original pixel resolutions of images in CVUSA vary between $12.7\frac{\text{cm}}{\text{px}}$ and $27.2\frac{\text{cm}}{\text{px}}$. Tab. 6.1 provides an overview of the varying side-lengths of aerial images in different evaluation settings.

For the first stage, we follow existing works and use images with the inconsistent scale provided by the datasets. Since the BEV-based matching requires a consistent scale, we resize all aerial images in the second stage to $16.8\frac{\text{cm}}{\text{px}}$ in VIGOR, $19.0\frac{\text{cm}}{\text{px}}$ in CVACT and $53.0\frac{\text{cm}}{\text{px}}$ in CVUSA, and crop or pad the result to 384×384 pixels.

Settings We evaluate our method in the four following settings based on whether the relative rotation and translation offset between street-view and aerial image is known or unknown.

CVUSA and CVACT represent a one-to-one matching task: Each aerial image is centered on the paired street-view image, and both images are north-aligned. We refer to this setting as *known translation* and *known orientation*. Since each aerial image covers only a single street-view camera pose, the retrieval hypothesis h is decomposed into just one pose hypotheses h', *i.e.* $f(h) = \{h'\}$.

We consider an additional setting where the street-view panorama is shifted randomly along the horizontal axis to simulate an *unknown orientation*. The pose hypotheses in $f(h)$ thus cover different possible orientations with the same center-aligned translation.

VIGOR represents a many-to-many matching task with a translation offset between street-view and aerial images. By default, panoramas are north-aligned as in CVUSA and CVACT. We refer to this setting as *unknown translation* and *known orientation*.

Analogous to the above case, we consider an additional setting in VIGOR with randomly shifted panoramas. This represents the most challenging of the four evaluation settings since $f(h)$ covers the largest subset of SE(2).

6.6 Evaluation

6.6.1 Implementation

This section provides details on the implementation of the method described in this chapter.

We train the model in both stages with the *Base* variant of ConvNeXt [80] pretrained on ImageNet [29]. The street-view features F_P are extracted at stride 8 with $c_P = 256$ channels. We use column-wise attention with 4 heads to project features from PV to BEV. The BEV map is constructed with $c_B = 32$ channels and a resolution of 48×48 in CVUSA and CVACT, and 19×19 in VIGOR. The aerial image is encoded at stride 8 with $c_A = 32$ channels and a resolution of 48×48. We use $|\Theta| = 32$ angles sampled uniformly in $[0, 2\pi)$ when evaluating with unknown orientation.

The first stage's model is trained for 40 epochs with a batch size of 128. We employ the Adam optimizer [63] with a learning rate of $1.0 \cdot 10^{-3}$, one epoch of linear warmup and the cosine decay schedule. The second stage's model is trained with a smaller batch size of 18 over 20 epochs with a learning rate of $1.0 \cdot 10^{-4}$ instead. The loss function in both stages is parametrized with a temperature of $\tau = 0.01$ and label smoothing of $\epsilon = 0.1$.

We employ the following data augmentation during training. In the case of *unknown orientation*, we shift the street-view panorama by a random angle in $[0, 2\pi)$, and rotate the aerial image by a random multiple of $\frac{\pi}{2}$ such that no parts of the image are cropped or padded. In the case of *known orientation*, we rotate the street-view and aerial images jointly to preserve the orientation alignment, and choose an angle as a random multiple of $\frac{\pi}{2}$. During the first stage, we randomly flip street-view and aerial images jointly [155].

We forward 100 candidates from the first to the second stage at test time.

6.6.2 Retrieval

Main results We report the recall of our two-stage method on the VIGOR, CVUSA and CVACT benchmarks with known and unknown orientation in Tab. 6.2 and Tab. 6.3. Our method outperforms previous works in all many-to-one settings, and achieves comparable results in one-to-one settings. The improvement is larger in more challenging settings, for instance when comparing *known orientation* with *unknown orientation*, or the cross-area and same-area splits in VIGOR. The largest improvement is demonstrated in the most challenging setting, *i.e.* the VIGOR cross-area split with *unknown orientation*. In this case, the second stage improves the recall from 31.2% to 65.0%. Fig. 6.3 shows examples of reranked images in the VIGOR dataset.

Cross-dataset results Tab. 6.4 reports results in the cross-dataset setting on CVUSA and CVACT proposed by Yang *et al.* [152] that measures the ability of the model to generalize from one dataset and geographical region to another.

First, we find that simply resizing aerial images in CVUSA and CVACT to a consistent pixel resolution without including our proposed second stage already improves recall by a significant amount. For instance, in the setting with *known orientation*, rescaling alone improves recall from 58.7% to 85.0% on CVUSA→CVACT and from 42.7% to 64.8% on CVACT→CVUSA. Tab. 6.1 shows that the pixel resolutions in both datasets diverge significantly, which results in a large domain gap when not adjusting for the inconsistent scale and explains the above observation.

We note that the ability of a model to generalize across scale differences is typically not required in a CVGL setting. Orthophotos are georegistered by definition and their scale is known, which allows adjusting the pixel resolution as required.

Second, employing our two-stage pipeline significantly boosts the recall, for instance from 55.3% to 78.1% on CVUSA→CVACT. When including our contribution of using a consistent scale, the improvement is even larger at 25.9% to 78.1%.

Ablation studies Tab. 6.5 shows the results of our ablation studies on the VIGOR dataset. We first test for the ideal resolution of the BEV map F_B by evaluating with the strides 4, 8 and 16 (*i.e.* $0.67\frac{\text{m}}{\text{px}}$, $1.34\frac{\text{m}}{\text{px}}$ and $2.68\frac{\text{m}}{\text{px}}$ for cells in F_B). We choose 128, 32 and 8 channels, such that the total number of features of F_B in the three tests is the same. The best recall is achieved with a stride of 8.

Tab. 6.2: Recall on the VIGOR dataset where the translational offset between street-view and aerial images is unknown. Baseline refers to the first stage of our method and is trained similar to Sample4Geo [30].

	Same-area				Cross-area			
	R@1	R@5	R@10	R@1%	R@1	R@5	R@10	R@1%
Known orientation								
SAFA [108]	33.93	58.42	68.12	98.24	8.20	19.59	26.36	77.61
TransGeo [160]	61.48	87.54	91.88	99.56	18.99	38.24	46.91	88.94
SAIG [165]	65.23	88.08	-	**99.68**	33.05	55.94	-	94.64
Sample4Geo [30]	77.86	95.66	97.21	99.61	61.70	83.50	88.00	**98.17**
Baseline	78.25	95.70	97.09	99.48	61.26	82.49	86.89	98.02
C-BEV (ours)	**87.49**	**97.65**	**98.26**	99.48	**80.01**	**92.02**	**93.33**	98.02
Unknown orientation								
TransGeo [160]	47.69	79.77	86.36	**99.29**	5.54	14.22	19.63	66.93
Baseline	66.09	90.73	93.38	98.31	31.14	52.11	59.41	**89.14**
C-BEV (ours)	**82.60**	**95.35**	**96.11**	98.31	**65.01**	**75.76**	**76.98**	**89.14**

Tab. 6.3: Recall on the CVUSA and CVACT datasets where aerial images are center-aligned with the paired street-view images. † denotes methods that use polar transformation on the aerial input image. Baseline refers to the first stage of our method and is trained similar to Sample4Geo [30].

	CVUSA				CVACT-Val				CVACT-Test			
	R@1	R@5	R@10	R@1%	R@1	R@5	R@10	R@1%	R@1	R@5	R@10	R@1%
Known orientation												
SAFA† [108]	89.84	96.93	98.14	99.64	81.03	92.80	94.84	98.17	-	-	-	-
DSM [110]	91.96	97.50	98.54	99.67	82.49	92.44	93.99	97.32	-	-	-	-
LPN [136]	92.83	98.00	98.85	99.78	83.66	94.14	95.92	98.41	-	-	-	-
TransGeo [160]	94.08	98.36	99.04	99.77	84.95	94.14	95.78	98.37	-	-	-	-
GeoDTR [155]	93.76	98.47	99.22	99.85	85.43	94.81	96.11	98.26	62.96	87.35	90.70	98.61
GeoDTR† [155]	95.43	98.86	99.34	99.86	86.21	95.44	96.72	98.77	64.52	88.59	91.96	98.74
SAIG [165]	96.08	98.72	99.22	99.86	89.21	96.07	97.04	98.74	-	-	-	-
SAIG† [165]	96.34	99.10	99.50	99.86	89.06	96.11	97.08	**98.89**	67.49	89.39	92.30	96.80
Sample4Geo [30]	98.68	99.68	99.78	**99.87**	90.81	**96.74**	**97.48**	98.77	71.51	92.42	94.45	98.70
Baseline	98.17	99.72	99.80	99.85	91.42	96.69	97.47	98.80	72.35	93.10	**94.92**	**98.77**
C-BEV (ours)	**98.72**	**99.77**	**99.83**	99.85	**91.68**	96.62	97.42	98.75	**75.94**	**93.23**	94.85	**98.77**
Unknown orientation												
DSM [110]	78.11	89.46	92.90	98.50	72.91	85.70	88.88	95.28	-	-	-	-
Baseline	92.09	97.73	98.32	99.26	86.95	93.34	94.63	97.16	68.10	88.18	90.53	**97.15**
C-BEV (ours)	**97.13**	**99.07**	**99.16**	**99.29**	**89.46**	**94.62**	**95.46**	**97.17**	**73.94**	**90.53**	**92.17**	**97.15**

Tab. 6.4: Results of the cross-dataset evaluation on CVUSA and CVACT as proposed by Yang *et al.* [152]. The model is trained on one dataset, and tested on the other dataset's validation split, respectively. † denotes methods that use polar transformation on the aerial input image. Baseline refers to the first stage of our method and is trained similar to Sample4Geo [30]. * indicates that images are resized to the same pixel resolution of $19.0\frac{\text{cm}}{\text{px}}$.

	CVUSA → CVACT				CVACT → CVUSA			
	R@1	R@5	R@10	R@1%	R@1	R@5	R@10	R@1%
Known orientation								
L2LTR† [152]	47.55	70.58	77.39	91.39	33.00	51.87	60.63	84.79
GeoDTR† [155]	53.16	75.62	81.90	93.80	44.07	64.66	72.08	90.09
Sample4Geo [30]	56.62	77.79	87.02	94.69	44.95	64.36	72.10	90.65
Baseline	58.73	82.35	87.65	96.41	42.72	64.99	72.97	92.45
Baseline*	85.02	95.50	96.68	**98.47**	64.77	79.91	84.85	96.09
C-BEV* (ours)	**88.07**	**96.15**	**97.07**	98.46	**70.01**	**83.57**	**87.82**	**96.25**
Unknown orientation								
Baseline	25.87	45.99	54.66	79.11	12.70	24.30	30.55	55.14
Baseline*	55.32	76.18	81.70	92.91	35.49	49.47	55.39	74.58
C-BEV* (ours)	**78.06**	**88.63**	**90.30**	**93.37**	**44.71**	**56.80**	**61.89**	**75.37**

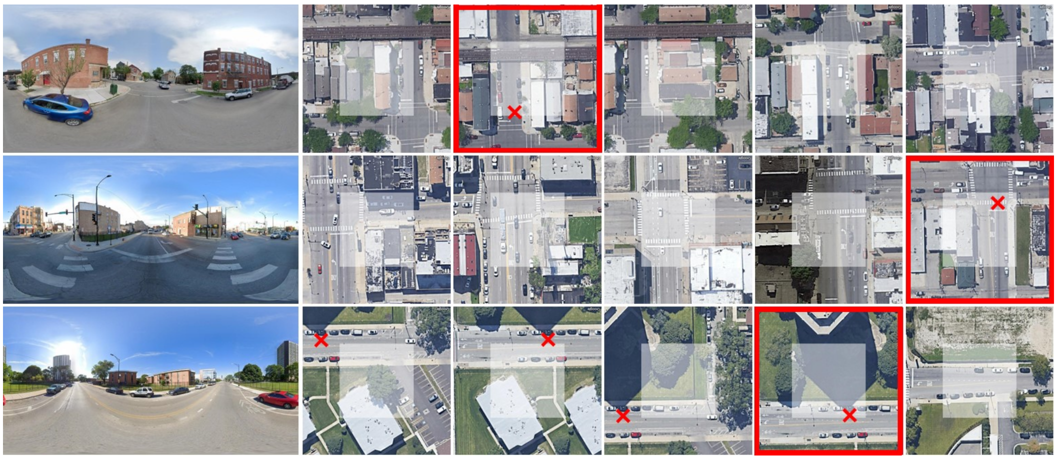


Fig. 6.3: Examples of the reranking in the second stage of our method applied to images from the VIGOR [163] dataset with unknown orientation. Each row presents a street-view query alongside the top 5 reference images retrieved in the first stage and ranked by descending matching score. A red border highlights the top-1 image after reranking, while a red cross marks the ground-truth camera position. White squares indicate the search regions for each image.

Tab. 6.5: Ablation studies for the image retrieval on the VIGOR dataset with known orientation. Our baseline is the first stage of our method and trained similar to Sample4Geo [30].

	Same-area		Cross-area	
	R@1	R@5	R@1	R@5
Baseline	78.25	95.70	61.26	82.49
BEV resolution				
Stride 4	86.69	97.50	78.67	91.59
Stride 8	**87.49**	**97.65**	80.01	**92.02**
Stride 16	86.85	97.59	**80.27**	91.80
First stage as prior				
With	**87.49**	**97.65**	**80.01**	**92.02**
Without	84.49	95.62	76.77	90.41
Number of candidates				
10	87.16	96.72	77.01	86.42
100	87.49	97.65	80.01	92.02
1000	**87.54**	**97.78**	**80.31**	**92.83**

Tab. 6.6: Pose error on matching image pairs in the VIGOR dataset. † refers to our method trained with a minimum distance of 50m between queries and negative reference images.

	Known orientation				Unknown orientation							
	Same-area		Cross-area		Same-area				Cross-area			
	Mean	Median	Mean	Median	Mean		Median		Mean		Median	
Supervised												
CVR [163]	8.99m	7.81m	8.89m	7.73m	-	-	-	-	-	-	-	-
MCC [146]	6.94m	3.64m	9.05m	5.14m	9.87m	56.86°	6.25m	16.02°	12.66m	72.13°	9.55m	29.97°
SliceMatch [71]	5.18m	2.58m	5.53m	2.55m	6.49m	25.46°	3.13m	4.71°	7.22m	25.97°	**3.31m**	4.51°
GGCVT [109]	4.12m	**1.34m**	5.16m	**1.40m**	-	-	-	-	-	-	-	-
Unsupervised												
C-BEV (ours)	**3.52m**	2.58m	**3.97m**	2.86m	**3.85m**	**5.55°**	**2.81m**	**2.16°**	**4.78m**	**7.90°**	3.48m	**2.52°**
C-BEV † (ours)	3.53m	2.70m	4.21m	3.31m	3.82m	5.64°	**2.81m**	2.20°	5.04m	8.24°	3.70m	2.66°

Next, we inspect the importance of including the scores computed in the first stage as prior probabilities in the second stage's reranking (*cf*. Eq. (6.6)). The model performance drops significantly when using only the second stage's scores. This highlights the importance of allowing the second stage's model to specialize on the failure cases of the first model, rather than rerank the given candidates from scratch.

Finally, we find that the recall improves when increasing the number of candidates forwarded to the second stage. We choose 100 candidates as a trade-off between recall and runtime.

6.6.3 Pose Estimation

Main results We report the results of the pose estimation in Tab. 6.6 and show example predictions in Fig. 6.4. Despite being trained in an unsupervised setting without metric ground-truth, our method achieves a lower mean translation and orientation error than existing methods trained in a supervised setting. The recent work GGCVT [109] achieves a higher mean error, but lower median error than our method. This indicates that C-BEV is more robust to outliers, but achieves a lower fine accuracy than GGCVT.

Search region boundary To investigate the ability of our model to learn pose estimation in an unsupervised manner, we consider the case of street-view images being located close to the search region boundary. Small offsets of the predicted camera location across the cell boundary onto a negative reference images are penalized in the loss function, if the negative reference image is included in the batch, and might therefore represent a direct form of metric supervision. To test whether such boundary cases are responsible for the model's pose estimation performance, we train the model by excluding negative reference images that are located closer than 50m to the query location. The resulting model performs only slightly worse (*cf*. Tab. 6.6), which indicates that accurate GNSS labels close to the search region boundary are not the primary cause of its ability to learn without metric supervision.

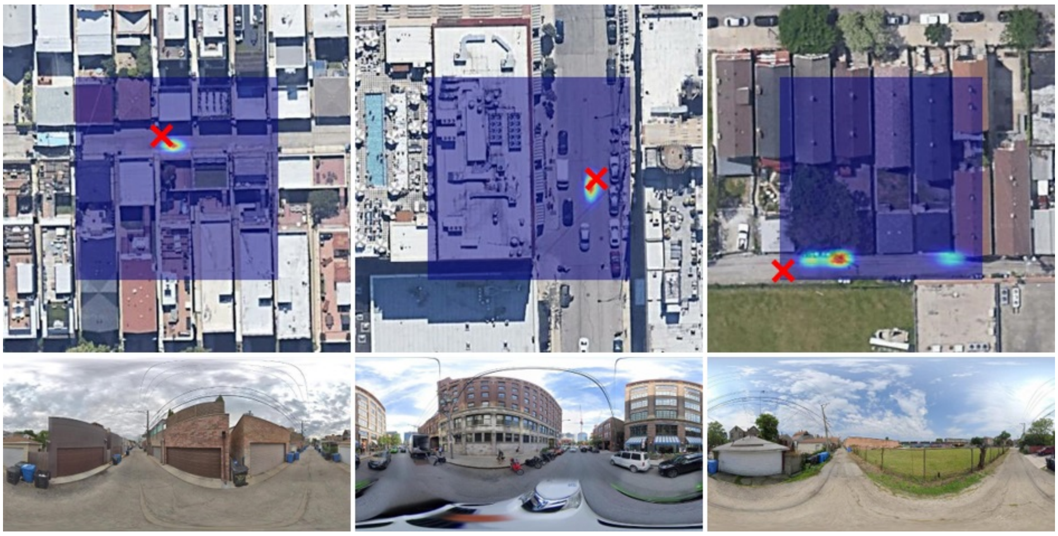


Fig. 6.4: Examples of the BEV-based, unsupervised pose estimation on the VIGOR [163] dataset with unknown orientation. *Bottom row:* Street-view panorama. *Top row:* Aerial image with predicted probabilities from low (blue) to high (red).

6.7 Summary

In this chapter, we present a novel view on CVGL that addresses the cross-view retrieval problem by representing it as an unsupervised pose estimation task. The method is integrated in a retrieve-and-rerank pipeline that significantly improves the recall of baseline methods, and is particularly effective in more challenging settings. The model further learns to predict camera poses in an unsupervised setting *despite not having access to ground-truth poses during training*, and even achieves competitive performance with recent supervised methods.

Our contributions are summarized as follows:

BEV-based retrieval We propose a novel method that utilizes the BEV-based matching presented in our pose estimation work (*cf*. Chapter 5) to address the retrieval problem. The method is based on a decomposition of a retrieval hypothesis into multiple pose estimation hypotheses, and a probabilistic fusion of the corresponding pose predictions. Employing the method in a retrieve-and-rerank pipeline significantly improves the recall of baseline methods. For instance, the recall of the state-of-the-art method Sample4Geo [30] on the VIGOR cross-area split [163] with unknown camera orientation is improved from 31.1% to 65.0% when including our proposed reranking stage.

Unsupervised pose estimation Due to the strong geometric bias in our proposed architecture, the model learns to accurately predict camera poses *despite being supervised only through a classical retrieval loss*. The model does not see ground-truth poses during training, but achieves pose predictions that are competitive with recent supervised approaches.

Scale inconsistency Our work is the first to point out and evaluate the scale inconsistency in existing benchmark datasets. Rescaling aerial images in these benchmarks to a consistent resolution already improves recall by a significant amount, for instance from 58.7% to 85.0% and from 42.7% to 64.8% on the CVUSA→CVACT and CVACT→CVUSA benchmarks, respectively.

Part III

Conclusion

Summary 7

In this thesis, we addressed the task of CVGL, *i.e.* localizing street-view photos by matching against a database of aerial imagery. Our aim was was to push the boundaries of scalability and accuracy while for the first time allowing CVGL to be used under real-world conditions and outside of controlled benchmark environments (*cf*. Sec. 1.2). Towards this end, we introduced novel methods, datasets and insight that allow localizing street-view photos in larger regions (**RQ1**), with higher fine-grained accuracy (**RQ2**) and under fewer constraints (**RQ3**) than previously possible.

At the center of our thesis are the methodological contributions that revisit the task of CVGL from the ground up and thereby allow for significant advancements in the state-of-the-art. In each chapter, we adopted a data-centric approach by designing *general-purpose* and *scalable* models and training schemes, and leveraging large amounts of data and compute. Our work demonstrates the benefit of this approach over the existing reliance on hand-crafted problem solutions or expert knowledge in the field CVGL; a similar observation has been made about a range of other fields by Sutton in his essay *The Bitter Lesson*[1].

We focused on laying the right foundations for a data-centric approach on cross-view retrieval in Chapter 4 by revisiting and proposing a novel problem formulation. This facilitated among others a new and revised usage of aerial imagery: Ensuring that images have a consistent pixel resolution minimizes the domain gap between train and test data that is not solved by scaling alone. The multiple LoD of aerial images further increase the amount of information that the model utilizes to address the localization objective, which reflects an upper bound on the performance that is achievable via scaling. We chose a model architecture that is based on general-purpose building blocks widely used in other domains, and demonstrated that it outperforms existing models designed specifically for CVGL. Additionally, we proposed a general-purpose HEM strategy that approximates larger batch sizes in contrastive settings by clustering samples into smaller batches based on their distance in the embedding space.

Our contributions for the first time allow localizing street-view photos

1. in state-sized search regions such as Massachusetts with 23000km^2 (**RQ1**),

[1] Sutton: "The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. [...] breakthrough progress eventually arrives by an [...] approach based on scaling computation by search and learning." [114]

2. without access to street-view data from the search region (**RQ3**),
3. under real-world conditions with consumer-grade devices (**RQ3**), and
4. in the wild, *i.e.* without requiring information about the camera's intrinsics, lens distortion or orientation during training or testing (**RQ3**).

Experimental results showed that our method is able to localize $\sim$60% of all (non-panoramic) photos uploaded to the crowd-sourcing platform Mapillary in the state of Massachusetts to within 50m of their ground-truth location, without access to street-view data from this region during training or testing. In contrast, existing works in the field of VGL address only smaller RoI up to 115km^2 in size with access to corresponding street-view images, and mostly rely on controlled benchmark environments that do not transfer to real-world applications.

To address the pose estimation problem in Chapter 5, we shifted away from widely used classical or hand-crafted feature matching approaches and the use of range-scanners. We proposed the first end-to-end trainable, vision-based model architecture that is able to benefit from the scaling of a data-centric approach, and achieve sub-meter accurate pose predictions. In Chapter 6, we showed that a variant of the model even learns to perform pose estimation without receiving explicit pose supervision during training.

Our contributions for the first time allow determining the geo-registered ego-pose and long-term ego-trajectory of a platform

1. using only aerial imagery as reference database (**RQ2**) without access to street-view data from the test region or test vehicle (**RQ3**),
2. in a purely vision-based manner without requiring range scanners such as lidar or radar (**RQ2**) or external signals such as GNSS,
3. across a range of environments such as urban and rural regions (**RQ3**), and
4. with sub-meter accuracy (**RQ2**).

The method achieves a median pose error of 0.87m on the Ford AV dataset, and mean trajectory error of 0.78m on the KITTI-360 dataset, without requiring training data from the test region or test vehicle. In contrast, all existing methods for cross-view pose estimation yield an error of at least 5.0m in more than 70% of predictions on Ford AV, despite their usage of training data from the test region. We found that these methods rely mainly on prior information on the aerial image such as the location of roads and other drivable area, rather than on cross-view matching.

To fully leverage the scalability of our proposed models, we presented new and large-scale datasets for cross-view retrieval and pose estimation. The datasets are large enough to be used in a semi-infinite or few-epoch setting, and allow scaling the training duration without overfitting on the same samples. They contain a diverse range of parameters, such as the depicted scene, image noise, illumination and camera parameters, which allowed obtaining robust models by simply relying on their ability to generalize over these parameters during training.

8 New Research Directions

This thesis has established CVGL as a viable method for VGL and opened up new directions for future research.

Our method for cross-view retrieval has surpassed existing limitations on the size of search regions by two orders of magnitude to cover entire states such as Massachusetts. However, further progress is necessary to rival the global availability of GNSS, and must address the *efficiency* of retrieval-based CVGL, for instance utilizing non-uniform search cells or hierarchical approaches. Our flexible online sampling of aerial images (*cf*. Fig. 4.10) allows for future research works to investigate how the RoI should be partitioned into cells, and how aerial images should be chosen to best predict an embedding for each cell.

This thesis has established CVGL as a viable competitor to the widely adapted SVGL, demonstrated its advantages in terms of scalability and coverage of regions without densely sampled street-view photos, and provided insight into their methodological similarities. This opens up new questions on how CVGL and SVGL might be integrated to provide a universal framework for VGL that benefits from the strengths of both types of reference data.

Our novel strategy for mining hard examples during training reformulates the objective of HEM as simply approximating the usage of larger batch sizes in contrastive settings. The approach is general-purpose and does not rely on domain-specific characteristics of CVGL. Future work might demonstrate its applicability to other domains such as SVGL or CLIP and investigate the full spectrum of clustering approaches to benefit methods for contrastive learning.

Addressing the long-tailed, non-uniform distribution of crowd-sourced datasets such as Mapillary presents another important challenge for future research. While our method utilizes a simple strategy of dividing data into geographic cells to mitigate this issue, more sophisticated data sampling could further improve performance. This includes for instance choosing a roughly uniform sampling strategy over other parameters such as time of day or season, or utilizing an unsupervised approach that samples data uniformly over the embedding space.

Our proposed approach for cross-view retrieval is able to localize individual street-view photos in large geographical regions. However, the recall of $\sim$60% suggests that images captured under real-world conditions such as those represented by

Mapillary might not always contain sufficient information to uniquely determine their geolocation. A possible option for future research to address the inherent ambiguity of individual photos is to perform retrieval on entire videos instead, which provides a much larger amount of information to the model for predicting matching embeddings between queries and references.

Finally, we have demonstrated the large benefits of shifting to scalable and data-centric methods for CVGL, where research (1) lays the right foundations via a suitable problem formulation, (2) utilizes general-purpose building blocks and (3) relies on *learning* rather than expert knowledge to find good solutions for the problem where possible.

Part IV

Appendix

Authored publications

The dissertation resulted in the following thesis-related publications:

1. Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens and Rainer Stiefelhagen
   Statewide Visual Geolocalization in the Wild
   In European Conference on Computer Vision (ECCV), 2024

2. Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens and Rainer Stiefelhagen
   Uncertainty-aware Vision-based Metric Cross-view Geolocalization
   In Conference on Computer Vision and Pattern Recognition (CVPR), 2023

3. Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens and Rainer Stiefelhagen
   C-BEV: Contrastive Bird's Eye View Training for Cross-View Image Retrieval and 3-DoF Pose Estimation
   In Computing Research Repository (CoRR), 2023

4. Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens and Rainer Stiefelhagen
   Continuous Self-localization on Aerial Images Using Visual and Lidar Sensors
   In International Conference on Intelligent Robots and Systems (IROS), 2022

The following publications were (co)authored by Florian Fervers, but are not directly related to the thesis content:

1. Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens and Rainer Stiefelhagen
   Improving Semantic Image Segmentation via Label Fusion in Semantically Textured Meshes
   In International Conference on Computer Vision Theory and Applications (VISAPP), 2022

2. Sebastian Bullinger, Florian Fervers, Christoph Bodensteiner, Michael Arens
   Geo-Tiles for Semantic Segmentation of Earth Observation Imagery
   In Computing Research Repository (CoRR), 2023

3. Philipp Fervers, Florian Fervers, Miriam Rinneburger, Mathilda Weisthoff, Jonathan Kottlors, Robert Reimer, David Zopfs, Erkan Celik, David Maintz, Nils Große-Hokamp, Thorsten Persigehl
   Physiological iodine uptake of the spine's bone marrow in dual-energy computed tomography – using artificial intelligence to define reference values based on 678 CT examinations of 189 individuals
   In Frontiers in Endocrinology, 2023

4. Philipp Fervers, Florian Fervers, Mathilda Weisthoff, Miriam Rinneburger, David Zopfs, Robert Peter Reimer, Gregor Pahn, Jonathan Kottlors, David Maintz, Simon Lennartz, Thorsten Persigehl, Nils Große Hokamp
   Dual-Energy CT, Virtual Non-Calcium Bone Marrow Imaging of the Spine: An AI-Assisted, Volumetric Evaluation of a Reference Cohort with 500 CT Scans
   In Diagnostics, 2022

5. Philipp Fervers, Florian Fervers, Astha Jaiswal, Miriam Rinneburger, Mathilda Weisthoff, Philip Pollmann-Schweckhorst, Jonathan Kottlors, Heike Carolus, Simon Lennartz, David Maintz, Rahil Shahzad, Thorsten Persigehl
   Assessment of COVID-19 lung involvement on computed tomography by deep-learning-, threshold-, and human reader-based approaches—an international, multi-center comparative study
   In Quantitative Imaging in Medicine and Surgery, 2022

6. Philipp Fervers, Florian Fervers, Jonathan Kottlors, Philipp Lohneis, Philip Pollman-Schweckhorst, Hasan Zaytoun, Miriam Rinneburger, David Maintz, Nils Große Hokamp
   Feasibility of artificial intelligence–supported assessment of bone marrow infiltration using dual-energy computed tomography in patients with evidence of monoclonal protein — a retrospective observational study
   In Computed Tomography, 2022

Bibliography

[1] *ADS-B Exchange*. `https://www.adsbexchange.com/`. [Online; accessed 21-August-2024] (cit. on p. 12).

[2] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. „Building Rome in a day“. In: *International Conference on Computer Vision*. 2009 (cit. on p. 13).

[3] Siddharth Agarwal, Ankit Vora, Gaurav Pandey, et al. „Ford multi-AV seasonal dataset“. In: *International Journal of Robotics Research* (2020) (cit. on pp. 16, 98, 102, 104, 105).

[4] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. „Freak: Fast retina keypoint“. In: *Conference on Computer Vision and Pattern Recognition*. 2012 (cit. on p. 19).

[5] Amar Ali-Bey, Brahim Chaib-Draa, and Philippe Giguere. „Mixvpr: Feature mixing for visual place recognition“. In: *Winter Conference on Applications of Computer Vision*. 2023 (cit. on p. 33).

[6] Amar Ali-bey, Brahim Chaib-draa, and Philippe Giguere. „Gsv-cities: Toward appropriate supervised visual place recognition“. In: *Neurocomputing* (2022) (cit. on p. 33).

[7] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. „NetVLAD: CNN architecture for weakly supervised place recognition“. In: *Conference on Computer Vision and Pattern Recognition*. 2016 (cit. on pp. 28, 32, 33, 57, 61, 72).

[8] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. „Layer normalization“. In: *Computing Research Repository* (2016) (cit. on pp. 57, 58).

[9] Mayank Bansal, Harpreet S Sawhney, Hui Cheng, and Kostas Daniilidis. „Geo-localization of street views with aerial image databases“. In: *ACM International Conference on Multimedia*. 2011 (cit. on pp. 19, 48).

[10] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. „Surf: Speeded up robust features“. In: *European Conference on Computer Vision*. 2006 (cit. on pp. 19, 48).

[11] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. „Curriculum learning“. In: *International Conference on Machine Learning*. 2009 (cit. on p. 63).

[12] Gabriele Berton, Carlo Masone, and Barbara Caputo. „Rethinking visual geo-localization for large-scale applications“. In: *Conference on Computer Vision and Pattern Recognition*. 2022 (cit. on pp. 9, 11, 13, 33, 48, 71, 72, 75).

[13] Gabriele Berton, Riccardo Mereu, Gabriele Trivigno, et al. „Deep visual geo-localization benchmark“. In: *Conference on Computer Vision and Pattern Recognition*. 2022 (cit. on p. 33).

[14] Gabriele Berton, Gabriele Trivigno, Barbara Caputo, and Carlo Masone. „Eigenplaces: Training viewpoint robust models for visual place recognition“. In: *International Conference on Computer Vision*. 2023 (cit. on p. 33).

[15] Anna Bosch, Andrew Zisserman, and Xavier Munoz. „Image classification using random forests and ferns“. In: *International Conference on Computer Vision*. Ieee. 2007, pp. 1–8 (cit. on p. 19).

[16] James Bradbury, Roy Frostig, Peter Hawkins, et al. *JAX: composable transformations of Python+NumPy programs*. 2018 (cit. on pp. 93, 118).

[17] Marcus A Brubaker, Andreas Geiger, and Raquel Urtasun. „Lost! leveraging the crowd for probabilistic visual self-localization“. In: *Conference on Computer Vision and Pattern Recognition*. 2013 (cit. on pp. 39, 42).

[18] Holger Caesar, Varun Bankiti, Alex H Lang, et al. „nuscenes: A multimodal dataset for autonomous driving“. In: *Conference on Computer Vision and Pattern Recognition*. 2020 (cit. on pp. 91, 98).

[19] Sudong Cai, Yulan Guo, Salman Khan, Jiwei Hu, and Gongjian Wen. „Ground-to-aerial image geo-localization with a hard exemplar reweighting triplet loss“. In: *International Conference on Computer Vision*. 2019 (cit. on p. 31).

[20] John Canny. „A computational approach to edge detection“. In: *Transactions on Pattern Analysis and Machine Intelligence* (1986) (cit. on p. 38).

[21] Nicholas Carlevaris-Bianco, Arash K Ushani, and Ryan M Eustice. „University of Michigan North Campus long-term vision and lidar dataset“. In: *International Journal of Robotics Research* (2016) (cit. on p. 72).

[22] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, et al. „Argoverse: 3d tracking and forecasting with rich maps“. In: *Conference on Computer Vision and Pattern Recognition*. 2019 (cit. on p. 98).

[23] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. „Return of the devil in the details: Delving deep into convolutional nets“. In: *Computing Research Repository* (2014) (cit. on pp. 28, 74).

[24] David M Chen, Georges Baatz, Kevin Köser, et al. „City-scale landmark identification on mobile devices“. In: *Conference on Computer Vision and Pattern Recognition*. 2011 (cit. on p. 72).

[25] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. „Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs“. In: *Transactions on Pattern Analysis and Machine Intelligence* (2017) (cit. on p. 89).

[26] Marius Cordts, Mohamed Omran, Sebastian Ramos, et al. „The cityscapes dataset for semantic urban scene understanding“. In: *Conference on Computer Vision and Pattern Recognition*. 2016 (cit. on p. 28).

[27] Igor Cvišić, Ivan Marković, and Ivan Petrović. „Soft2: Stereo visual odometry for road vehicles based on a point-to-epipolar-line metric“. In: *Transactions on Robotics* (2022) (cit. on p. 108).

[28] Pierre Dellenbach, Jean-Emmanuel Deschaud, Bastien Jacquet, and François Goulette. „Ct-icp: Real-time elastic lidar odometry with loop closure“. In: *International Conference on Robotics and Automation*. 2022 (cit. on p. 108).

[29] Jia Deng, Wei Dong, Richard Socher, et al. „Imagenet: A large-scale hierarchical image database“. In: *Conference on Computer Vision and Pattern Recognition*. 2009 (cit. on pp. 28, 73, 88, 103, 120).

[30] Fabian Deuser, Konrad Habel, and Norbert Oswald. „Sample4geo: Hard negative sampling for cross-view geo-localisation“. In: *International Conference on Computer Vision*. 2023 (cit. on pp. 11, 18, 26, 28, 30–32, 48, 57, 59, 61, 74, 76, 79, 80, 116, 118, 122, 123, 125).

[31] Shengyong Ding, Liang Lin, Guangrun Wang, and Hongyang Chao. „Deep feature learning with relative distance comparison for person re-identification“. In: *Pattern Recognition* (2015) (cit. on p. 30).

[32] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. „An image is worth 16x16 words: Transformers for image recognition at scale“. In: *Computing Research Repository* (2020) (cit. on pp. 28, 56, 58).

[33] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, et al. „The Faiss library“. In: *Computing Research Repository* (2024) (cit. on pp. 74, 76, 84).

[34] Leonhard Euler. *De repraesentatione superficiei sphaericae super plano*. 1778 (cit. on p. 27).

[35] Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens, and Rainer Stiefelhagen. „C-BEV: Contrastive Bird’s Eye View Training for Cross-View Image Retrieval and 3-DoF Pose Estimation“. In: *Computing Research Repository* (2023) (cit. on pp. 18, 26, 28, 67, 114).

[36] Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens, and Rainer Stiefelhagen. „Continuous self-localization on aerial images using visual and lidar sensors“. In: *International Conference on Intelligent Robots and Systems*. 2022 (cit. on pp. 17, 86).

[37] Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens, and Rainer Stiefelhagen. „Statewide Visual Geolocalization in the Wild“. In: *European Conference on Computer Vision*. 2024 (cit. on pp. 17, 49).

[38] Florian Fervers, Sebastian Bullinger, Christoph Bodensteiner, Michael Arens, and Rainer Stiefelhagen. „Uncertainty-aware vision-based metric cross-view geolocalization“. In: *Conference on Computer Vision and Pattern Recognition*. 2023 (cit. on pp. 17, 86).

[39] Martin A Fischler and Robert C Bolles. „Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography“. In: *Communications of the ACM* (1981) (cit. on p. 115).

[40] Georgios Floros, Benito Van Der Zander, and Bastian Leibe. „Openstreetslam: Global vehicle localization using openstreetmaps“. In: *International Conference on Robotics and Automation*. 2013 (cit. on pp. 39, 42, 43).

[41] Philip Gage. „A new algorithm for data compression“. In: *The C Users Journal* (1994) (cit. on p. 58).

[42] *Geobasis BB*. `https://data.geobasis-bb.de/geobasis/daten/dop/rgb_jpg/`. [Online; accessed 21-August-2024] (cit. on pp. 64, 65, 68, 73).

[43] *Geodaten Sachsen*. `https://www.geodaten.sachsen.de/luftbild-produkte-3995.html`. [Online; accessed 21-August-2024] (cit. on pp. 64, 65, 68, 73).

[44] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, 2016 (cit. on pp. 30, 73).

[45] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, et al. „Generative adversarial nets“. In: *Advances in Neural Information Processing Systems* (2014) (cit. on p. 31).

[46] Priya Goyal, Piotr Dollár, Ross Girshick, et al. „Accurate, large minibatch sgd: Training imagenet in 1 hour“. In: *Computing Research Repository* (2017) (cit. on p. 74).

[47] Giorgio Grisetti, Rainer Kümmerle, Hauke Strasdat, and Kurt Konolige. „g2o: A general framework for (hyper) graph optimization“. In: *International Conference on Robotics and Automation*. 2011 (cit. on pp. 100, 101).

[48] Silviu Guiaşu. *Information theory with applications*. 1977 (cit. on p. 38).

[49] Raia Hadsell, Sumit Chopra, and Yann LeCun. „Dimensionality reduction by learning an invariant mapping“. In: *Conference on Computer Vision and Pattern Recognition*. 2006 (cit. on pp. 20, 29).

[50] Adam W Harley, Zhaoyuan Fang, Jie Li, Rares Ambrus, and Katerina Fragkiadaki. „A simple baseline for bev perception without lidar“. In: *Computing Research Repository* (2022) (cit. on p. 85).

[51] Stephen Hausler, Sourav Garg, Ming Xu, Michael Milford, and Tobias Fischer. „Patch-netvlad: Multi-scale fusion of locally-global descriptors for place recognition“. In: *Conference on Computer Vision and Pattern Recognition*. 2021 (cit. on pp. 33, 114, 115).

[52] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. „Deep residual learning for image recognition“. In: *Conference on Computer Vision and Pattern Recognition*. 2016 (cit. on pp. 19, 56, 57, 89).

[53] Ruining He, Anirudh Ravula, Bhargav Kanagal, and Joshua Ainslie. „Realformer: Transformer likes residual attention“. In: *Computing Research Repository* (2020) (cit. on p. 92).

[54] Dan Hendrycks and Kevin Gimpel. „Gaussian error linear units (gelus)“. In: *Computing Research Repository* (2016) (cit. on p. 57).

[55] Alexander Hermans, Lucas Beyer, and Bastian Leibe. „In defense of the triplet loss for person re-identification“. In: *Computing Research Repository* (2017) (cit. on pp. 30, 32, 61).

[56] John Houston, Guido Zuidhof, Luca Bergamini, et al. „One thousand and one hours: Self-driving motion prediction dataset“. In: *Computing Research Repository* (2020) (cit. on pp. 98, 106).

[57] Sixing Hu, Mengdan Feng, Rang MH Nguyen, and Gim Hee Lee. „Cvm-net: Cross-view matching network for image-based ground-to-aerial geo-localization“. In: *Conference on Computer Vision and Pattern Recognition*. 2018 (cit. on pp. 28, 106).

[58] Mahdi Javanmardi, Ehsan Javanmardi, Yanlei Gu, and Shunsuke Kamijo. „Towards high-definition 3D urban mapping: Road feature-based registration of mobile mapping systems and aerial imagery“. In: *Remote Sensing* (2017) (cit. on pp. 38, 42).

[59] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. „Aggregating local descriptors into a compact image representation“. In: *Conference on Computer Vision and Pattern Recognition*. 2010 (cit. on p. 9).

[60] Rudolph Emil Kalman. „A new approach to linear filtering and prediction problems“. In: *Journal of Basic Engineering* (1960) (cit. on p. 96).

[61] Hyo Jin Kim, Enrique Dunn, and Jan-Michael Frahm. „Learned Contextual Feature Reweighting for Image Geo-Localization.“ In: *Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on p. 33).

[62] Jonghwi Kim and Jinwhan Kim. „Fusing lidar data and aerial imagery with perspective correction for precise localization in urban canyons“. In: *International Conference on Intelligent Robots and Systems*. 2019 (cit. on pp. 38, 42, 43).

[63] Diederik P Kingma. „Adam: A method for stochastic optimization“. In: *Computing Research Repository* (2014) (cit. on pp. 73, 120).

[64] S. Kocherlakota and K. Kocherlakota. „Generalized Variance“. In: *Encyclopedia of Statistical Sciences*. John Wiley & Sons, 2004 (cit. on p. 102).

[65] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. „Imagenet classification with deep convolutional neural networks“. In: *Advances in Neural Information Processing Systems* (2012) (cit. on pp. 19, 28).

[66] Rainer Kümmerle, Bastian Steder, Christian Dornhege, et al. „Large scale graph-based SLAM using aerial images as prior information“. In: *Autonomous Robots* (2011) (cit. on pp. 38, 43).

[67] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. „Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks“. In: *International Conference on Machine Learning*. 2021 (cit. on p. 74).

[68] John Lambert. *The Bayes Filter and Intro to State Estimation*. `https://johnwlambert.github.io/bayes-filter/`. [Online; accessed 21-August-2024] (cit. on p. 95).

[69] Juho Lee, Yoonho Lee, Jungtaek Kim, et al. „Set transformer: A framework for attention-based permutation-invariant neural networks“. In: *International Conference on Machine Learning*. 2019 (cit. on pp. 55, 56).

[70] Alfred Leick, Lev Rapoport, and Dmitry Tatarnikov. *GPS satellite surveying*. John Wiley & Sons, 2015 (cit. on p. 11).

[71] Ted Lentsch, Zimin Xia, Holger Caesar, and Julian FP Kooij. „Slicematch: Geometry-guided aggregation for cross-view pose estimation“. In: *Conference on Computer Vision and Pattern Recognition*. 2023 (cit. on pp. 27, 67, 124).

[72] Guopeng Li, Ming Qian, and Gui-Song Xia. „Unleashing Unlabeled Data: A Paradigm for Cross-View Geo-Localization“. In: *Conference on Computer Vision and Pattern Recognition*. 2024 (cit. on pp. 25, 26).

[73] Zhiqi Li, Wenhai Wang, Hongyang Li, et al. „Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers“. In: *European Conference on Computer Vision*. 2022 (cit. on p. 91).

[74] Yiyi Liao, Jun Xie, and Andreas Geiger. „KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d“. In: *Computing Research Repository* (2021) (cit. on pp. 98, 108).

[75] Tsung-Yi Lin, Yin Cui, Serge Belongie, and James Hays. „Learning deep representations for ground-to-aerial geolocalization“. In: *Conference on Computer Vision and Pattern Recognition*. 2015 (cit. on pp. 19, 20, 28, 29).

[76] Tsung-Yi Lin, Michael Maire, Serge Belongie, et al. „Microsoft coco: Common objects in context“. In: *European Conference on Computer Vision*. 2014 (cit. on p. 28).

[77] Liu Liu and Hongdong Li. „Lending orientation to neural networks for cross-view geo-localization“. In: *Conference on Computer Vision and Pattern Recognition*. 2019 (cit. on pp. 11, 21–23, 25, 27, 72, 119).

[78] Liyuan Liu, Haoming Jiang, Pengcheng He, et al. „On the variance of the adaptive learning rate and beyond“. In: *Computing Research Repository* (2019) (cit. on p. 103).

[79] Zhijian Liu, Haotian Tang, Alexander Amini, et al. „Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation“. In: *International Conference on Robotics and Automation*. 2023 (cit. on p. 85).

[80] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, et al. „A convnet for the 2020s“. In: *Conference on Computer Vision and Pattern Recognition*. 2022 (cit. on pp. 28, 47, 55, 56, 73, 88, 89, 103, 116, 120).

[81] Ilya Loshchilov and Frank Hutter. „Decoupled weight decay regularization“. In: *Computing Research Repository* (2017) (cit. on pp. 73, 103).

[82] David G Lowe. „Distinctive image features from scale-invariant keypoints“. In: *International Journal of Computer Vision* (2004) (cit. on pp. 19, 48).

[83] Yuexin Ma, Tai Wang, Xuyang Bai, et al. „Vision-centric bev perception: A survey“. In: *Computing Research Repository* (2022) (cit. on p. 40).

[84] Yu A Malkov and Dmitry A Yashunin. „Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs“. In: *Transactions on Pattern Analysis and Machine Intelligence* (2018) (cit. on p. 74).

[85] *Mapillary*. `https://www.mapillary.com/`. [Online; accessed 21-August-2024] (cit. on p. 67).

[86] *MassGIS*. `https://www.mass.gov/info-details/massgis-data-2021-aerial-imagery`. [Online; accessed 21-August-2024] (cit. on pp. 10, 11, 52, 64, 65, 68, 77, 99).

[87] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. „Robust wide-baseline stereo from maximally stable extremal regions“. In: *Image and Vision Computing* (2004) (cit. on p. 19).

[88] Michael J Milford and Gordon F Wyeth. „Mapping a suburb with a single camera using a biologically inspired SLAM system“. In: *Transactions on Robotics* (2008) (cit. on p. 72).

[89] Ian D Miller, Anthony Cowley, Ravi Konkimalla, et al. „Any way you look at it: Semantic crossview localization and mapping with lidar“. In: *Robotics and Automation Letters* (2021) (cit. on pp. 39, 43).

[90] Roi Mit, Yoav Zangvil, and Dror Katalan. „Analyzing tesla‘s level 2 autonomous driving system under different gnss spoofing scenarios and implementing connected services for authentication and reliability of gnss data“. In: *International Technical Meeting of the Satellite Division of The Institute of Navigation*. 2020 (cit. on p. 13).

[91] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. „When does label smoothing help?“ In: *Advances in Neural Information Processing Systems* (2019) (cit. on p. 94).

[92] *NC OneMap*. `https://www.nconemap.gov/`. [Online; accessed 21-August-2024] (cit. on pp. 64, 65, 68).

[93] Masafumi Noda, Tomokazu Takahashi, Daisuke Deguchi, et al. „Vehicle ego-localization by matching in-vehicle camera images to an aerial image“. In: *Asian Conference on Computer Vision Workshops*. 2011 (cit. on p. 40).

[94] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. „Representation learning with contrastive predictive coding“. In: *Computing Research Repository* (2018) (cit. on pp. 30, 32, 59).

[95] *OpenData DC*. `https://opendata.dc.gov/datasets/DCGIS::aerial-photography-orthophoto-2021/about`. [Online; accessed 21-August-2024] (cit. on pp. 64, 65, 68, 99).

[96] *OpenGeoData NRW*. `https://www.opengeodata.nrw.de/produkte/geobasis/lusat/akt/dop/`. [Online; accessed 21-August-2024] (cit. on pp. 10, 64, 65, 68).

[97] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. „On the difficulty of training recurrent neural networks“. In: *International Conference on Machine Learning*. 2013 (cit. on p. 73).

[98] Adam Paszke, Sam Gross, Francisco Massa, et al. „Pytorch: An imperative style, high-performance deep learning library“. In: *Advances in Neural Information Processing Systems* (2019) (cit. on pp. 93, 118).

[99] Lang Peng, Zhirong Chen, Zhangjie Fu, Pengpeng Liang, and Erkang Cheng. „Bevsegformer: Bird’s eye view semantic segmentation from arbitrary camera rigs“. In: *Winter Conference on Applications of Computer Vision*. 2023 (cit. on p. 41).

[100] Oliver Pink. „Visual map matching and localization using a global feature map“. In: *Conference on Computer Vision and Pattern Recognition Workshops*. 2008 (cit. on pp. 38, 40, 42).

[101] Alec Radford, Jong Wook Kim, Chris Hallacy, et al. „Learning transferable visual models from natural language supervision“. In: *International Conference on Machine Learning*. 2021 (cit. on pp. 30, 31, 59–61, 118).

[102] Krishna Regmi and Mubarak Shah. „Bridging the domain gap for ground-to-aerial image matching“. In: *International Conference on Computer Vision*. 2019 (cit. on p. 31).

[103] Dávid Rozenberszki and András L Majdik. „LOL: Lidar-only Odometry and Localization in 3D point cloud maps“. In: *International Conference on Robotics and Automation*. 2020 (cit. on p. 85).

[104] Johannes L Schonberger and Jan-Michael Frahm. „Structure-from-motion revisited“. In: *Conference on Computer Vision and Pattern Recognition*. 2016 (cit. on p. 12).

[105] Florian Schroff, Dmitry Kalenichenko, and James Philbin. „Facenet: A unified embedding for face recognition and clustering“. In: *Conference on Computer Vision and Pattern Recognition*. 2015 (cit. on p. 29).

[106] Paul Hongsuck Seo, Tobias Weyand, Jack Sim, and Bohyung Han. „Cplanet: Enhancing image geolocalization by combinatorial partitioning of maps“. In: *European Conference on Computer Vision*. 2018 (cit. on p. 34).

[107] Yujiao Shi and Hongdong Li. „Beyond cross-view image retrieval: Highly accurate vehicle localization using satellite image“. In: *Conference on Computer Vision and Pattern Recognition*. 2022 (cit. on pp. 11, 40–42, 99, 102–106).

[108] Yujiao Shi, Liu Liu, Xin Yu, and Hongdong Li. „Spatial-aware feature aggregation for image based cross-view geo-localization“. In: *Advances in Neural Information Processing Systems* (2019) (cit. on pp. 25, 28, 31, 48, 57, 74, 76, 79, 106, 122).

[109] Yujiao Shi, Fei Wu, Akhil Perincherry, Ankit Vora, and Hongdong Li. „Boosting 3-dof ground-to-satellite camera localization accuracy via geometry-guided cross-view transformer“. In: *International Conference on Computer Vision*. 2023 (cit. on p. 124).

[110] Yujiao Shi, Xin Yu, Dylan Campbell, and Hongdong Li. „Where am i looking at? joint location and orientation estimation by cross-view matching“. In: *Conference on Computer Vision and Pattern Recognition*. 2020 (cit. on pp. 106, 122).

[111] Yujiao Shi, Xin Yu, Shan Wang, and Hongdong Li. „Cvlnet: Cross-view semantic correspondence learning for video-based camera localization“. In: *Asian Conference on Computer Vision*. 2022 (cit. on p. 27).

[112] D Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. John Wiley & Sons, 2006 (cit. on p. 96).

[113] *Stratmap*. `https://tnris.org/stratmap/`. [Online; accessed 21-August-2024] (cit. on p. 99).

[114] Richard S. Sutton. *The Bitter Lesson*. [Online; accessed 21-August-2024]. 2019 (cit. on p. 129).

[115] Daniel Svensson. „Derivation of the discrete-time constant turn rate and acceleration motion model“. In: *Sensor Data Fusion: Trends, Solutions, Applications*. 2019 (cit. on p. 95).

[116] Tim Y Tang, Daniele De Martini, and Paul Newman. „Get to the point: Learning lidar place recognition and metric localisation using overhead imagery“. In: *Robotics: Science and Systems* (2021) (cit. on pp. 38, 42).

[117] Tim Y Tang, Daniele De Martini, Shangzhe Wu, and Paul Newman. „Self-supervised learning for using overhead imagery as maps in outdoor range sensor localization“. In: *International Journal of Robotics Research* (2021) (cit. on pp. 38, 42).

[118] Tim Yuqing Tang, Daniele De Martini, Dan Barnes, and Paul Newman. „Rsl-net: Localising in satellite images from a radar on the ground“. In: *Robotics and Automation Letters* (2020) (cit. on pp. 38, 42).

[119] Aysim Toker, Qunjie Zhou, Maxim Maximov, and Laura Leal-Taixé. „Coming down to earth: Satellite-to-street view synthesis for geo-localization“. In: *Conference on Computer Vision and Pattern Recognition*. 2021 (cit. on p. 31).

[120] Akihiko Torii, Relja Arandjelovic, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. „24/7 place recognition by view synthesis“. In: *Conference on Computer Vision and Pattern Recognition*. 2015 (cit. on p. 72).

[121] Akihiko Torii, Josef Sivic, Tomas Pajdla, and Masatoshi Okutomi. „Visual place recognition with repetitive structures“. In: *Conference on Computer Vision and Pattern Recognition*. 2013 (cit. on p. 9).

[122] Gabriele Trivigno, Gabriele Berton, Juan Aragon, Barbara Caputo, and Carlo Masone. „Divide&Classify: Fine-Grained Classification for City-Wide Visual Place Recognition“. In: *International Conference on Computer Vision*. 2023 (cit. on pp. 34, 35, 75).

[123] Frank Van Diggelen and Per Enge. „The world’s first GPS MOOC and worldwide laboratory using smartphones“. In: *International Technical Meeting of the Satellite Division of The Institute of Navigation*. 2015 (cit. on p. 11).

[124] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. „Attention is all you need“. In: *Advances in Neural Information Processing Systems* (2017) (cit. on pp. 28, 41, 55, 56, 58, 88, 90, 93).

[125] Lucas De Paula Veronese, Edilson De Aguiar, Rafael Correia Nascimento, et al. „Re-emission and satellite aerial maps applied to vehicle localization on urban environments“. In: *International Conference on Intelligent Robots and Systems*. 2015 (cit. on p. 39).

[126] Anirudh Viswanathan, Bernardo R Pires, and Daniel Huber. „Vision based robot localization by ground to satellite matching in gps-denied situations“. In: *International Conference on Intelligent Robots and Systems*. 2014 (cit. on pp. 19, 48).

[127] Anirudh Viswanathan, Bernardo R Pires, and Daniel Huber. „Vision-based robot localization across seasons and in remote locations“. In: *International Conference on Robotics and Automation*. 2016 (cit. on pp. 39, 43).

[128] Vicente Vivanco Cepeda, Gaurav Kumar Nayak, and Mubarak Shah. „Geoclip: Clip-inspired alignment between locations and images for effective worldwide geo-localization“. In: *Advances in Neural Information Processing Systems* (2024) (cit. on p. 34).

[129] Nam Vo, Nathan Jacobs, and James Hays. „Revisiting im2gps in the deep learning era“. In: *International Conference on Computer Vision*. 2017 (cit. on p. 34).

[130] Nam N Vo and James Hays. „Localizing and orienting street views using overhead imagery“. In: *European Conference on Computer Vision*. 2016 (cit. on pp. 22, 23, 27–31, 72).

[131] Ankit Vora, Siddharth Agarwal, Gaurav Pandey, and James McBride. „Aerial imagery based lidar localization for autonomous vehicles“. In: *Computing Research Repository* (2020) (cit. on pp. 39, 43).

[132] Shruti Vyas, Chen Chen, and Mubarak Shah. „Gama: Cross-view video geo-localization“. In: *European Conference on Computer Vision*. 2022 (cit. on pp. 22, 27, 72).

[133] Olga Vysotska and Cyrill Stachniss. „Improving SLAM by exploiting building information from publicly available maps and localization priors“. In: *Journal of Photogrammetry, Remote Sensing and Geoinformation Science* (2017) (cit. on pp. 37, 42, 43).

[134] Ruotong Wang, Yanqing Shen, Weiliang Zuo, Sanping Zhou, and Nanning Zheng. „Transvpr: Transformer-based place recognition with multi-level attention aggregation“. In: *Conference on Computer Vision and Pattern Recognition*. 2022 (cit. on pp. 114, 115).

[135] Shan Wang, Yanhao Zhang, Ankit Vora, Akhil Perincherry, and Hengdong Li. „Satellite image based cross-view localization for autonomous vehicle“. In: *International Conference on Robotics and Automation*. 2023 (cit. on p. 99).

[136] Tingyu Wang, Zhedong Zheng, Chenggang Yan, et al. „Each part matters: Local patterns facilitate cross-view geo-localization“. In: *Transactions on Circuits and Systems for Video Technology* (2021) (cit. on p. 122).

[137] Xin Wang, Yudong Chen, and Wenwu Zhu. „A survey on curriculum learning“. In: *Transactions on Pattern Analysis and Machine Intelligence* (2021) (cit. on p. 63).

[138] Xipeng Wang, Steve Vozar, and Edwin Olson. „Flag: Feature-based localization between air and ground“. In: *International Conference on Robotics and Automation*. 2017 (cit. on pp. 38, 43).

[139] Frederik Warburg, Soren Hauberg, Manuel Lopez-Antequera, et al. „Mapillary street-level sequences: A dataset for lifelong place recognition“. In: *Conference on Computer Vision and Pattern Recognition*. 2020 (cit. on pp. 32, 61).

[140] *Web Mercator Projection: EPSG:3857*. `https://epsg.io/3857`. [Online; accessed 21-August-2024] (cit. on pp. 27, 52).

[141] Eric W. Weisstein. *Convolution Theorem. From MathWorld—A Wolfram Web Resource*. `http://mathworld.wolfram.com/ConvolutionTheorem.html` (cit. on p. 93).

[142] Ross Wightman. *PyTorch Image Models*. `https://github.com/rwightman/pytorch-image-models`. 2019 (cit. on p. 104).

[143] Benjamin Wilson, William Qi, Tanmay Agarwal, et al. „Argoverse 2: Next generation datasets for self-driving perception and forecasting“. In: *Computing Research Repository* (2023) (cit. on pp. 98, 100).

[144] Scott Workman, Richard Souvenir, and Nathan Jacobs. „Wide-area image geolocalization with aerial reference imagery“. In: *International Conference on Computer Vision*. 2015 (cit. on pp. 11, 19, 21, 25, 27, 28, 72, 119).

[145] Zimin Xia, Olaf Booij, Marco Manfredi, and Julian FP Kooij. „Cross-view matching for vehicle localization by learning geographically local representations“. In: *Robotics and Automation Letters* (2021) (cit. on p. 41).

[146] Zimin Xia, Olaf Booij, Marco Manfredi, and Julian FP Kooij. „Visual cross-view metric localization with dense uncertainty estimates“. In: *European Conference on Computer Vision*. 2022 (cit. on pp. 41, 124).

[147] Pengchuan Xiao, Zhenlei Shao, Steven Hao, et al. „PandaSet: Advanced Sensor Suite Dataset for Autonomous Driving“. In: *Intelligent Transportation Systems Conference*. 2021 (cit. on p. 98).

[148] Enze Xie, Wenhai Wang, Zhiding Yu, et al. „SegFormer: Simple and efficient design for semantic segmentation with transformers“. In: *Advances in Neural Information Processing Systems* (2021) (cit. on p. 92).

[149] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. „Aggregated residual transformations for deep neural networks“. In: *Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on p. 56).

[150] Yifan Xu, Pourya Shamsolmoali, Eric Granger, et al. „TransVLAD: Multi-scale attention-based global descriptors for visual geo-localization“. In: *Winter Conference on Applications of Computer Vision*. 2023 (cit. on p. 33).

[151] Fan Yan, Olga Vysotska, and Cyrill Stachniss. „Global localization on openstreetmap using 4-bit semantic descriptors“. In: *European Conference on Mobile Robots*. 2019 (cit. on pp. 39, 43).

[152] Hongji Yang, Xiufan Lu, and Yingying Zhu. „Cross-view geo-localization with layer-to-layer transformer“. In: *Advances in Neural Information Processing Systems* (2021) (cit. on pp. 28, 119, 121, 122).

[153] Dongqiangzi Ye, Zixiang Zhou, Weijia Chen, et al. „Lidarmultinet: Towards a unified multi-task network for lidar perception“. In: *AAAI Conference on Artificial Intelligence*. 2023 (cit. on p. 85).

[154] Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, et al. „Decoupled contrastive learning“. In: *European Conference on Computer Vision*. 2022 (cit. on p. 59).

[155] Xiaohan Zhang, Xingyu Li, Waqas Sultani, Yi Zhou, and Safwan Wshah. „Cross-view geo-localization via learning disentangled geometric layout correspondence“. In: *AAAI Conference on Artificial Intelligence*. 2023 (cit. on pp. 27, 28, 72, 121, 122).

[156] Xiaohan Zhang, Waqas Sultani, and Safwan Wshah. „Cross-view image sequence geo-localization“. In: *Winter Conference on Applications of Computer Vision*. 2023 (cit. on pp. 22, 27).

[157] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. „Pyramid scene parsing network“. In: *Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on p. 89).

[158] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. „Unpaired image-to-image translation using cycle-consistent adversarial networks“. In: *International Conference on Computer Vision*. 2017 (cit. on p. 26).

[159] Minzhao Zhu, Yi Yang, Wenjie Song, Meiling Wang, and Mengyin Fu. „AGCV-LOAM: Air-ground cross-view based LiDAR odometry and mapping“. In: *Chinese Control and Decision Conference*. 2020 (cit. on pp. 40, 43).

[160] Sijie Zhu, Mubarak Shah, and Chen Chen. „Transgeo: Transformer is all you need for cross-view image geo-localization“. In: *Conference on Computer Vision and Pattern Recognition*. 2022 (cit. on pp. 26, 28, 31–33, 48, 61, 74, 76, 122).

[161] Sijie Zhu, Linjie Yang, Chen Chen, et al. „R2former: Unified retrieval and reranking transformer for place recognition“. In: *Conference on Computer Vision and Pattern Recognition*. 2023 (cit. on pp. 114, 115).

[162] Sijie Zhu, Taojiannan Yang, and Chen Chen. „Revisiting street-to-aerial view image geo-localization and orientation estimation“. In: *Winter Conference on Applications of Computer Vision*. 2021 (cit. on pp. 27, 33, 61).

[163] Sijie Zhu, Taojiannan Yang, and Chen Chen. „Vigor: Cross-view image geo-localization beyond one-to-one retrieval“. In: *Conference on Computer Vision and Pattern Recognition*. 2021 (cit. on pp. 11, 13, 18, 23, 24, 27, 31, 33, 34, 41, 48, 50, 52, 54, 61, 67, 71, 72, 74, 75, 79, 106, 113, 119, 123–125).

[164] Xizhou Zhu, Weijie Su, Lewei Lu, et al. „Deformable detr: Deformable transformers for end-to-end object detection“. In: *Computing Research Repository* (2020) (cit. on p. 91).

[165] Yingying Zhu, Hongji Yang, Yuxin Lu, and Qiang Huang. „Simple, effective and general: A new backbone for cross-view image geo-localization“. In: *Computing Research Repository* (2023) (cit. on pp. 28, 30, 32, 57, 79, 122).

Colophon

This thesis was typeset with $\LaTeX\,2_\varepsilon$. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.